The Computer Bulletin

Volume 16 Number 6 June 1972

The Computer Bulletin

Editor Graham Weiner BSc (Eng), MBCS

Chairman of Editorial Board
T. F. Goodwin, C Eng, AFRAeS, FBCS

Editorial Office 29 Portland Place, London W1N 4AP Tel: 01-637 0471

The Computer Bulletin is published monthly and issued free to all members of The British Computer Society. The subscription rate to non-members is £4·50 per annum, post paid. Single copies cost 50p.

Back issues of *The Computer Bulletin* are handled by Messrs Wm. Dawson & Sons Limited, Cannon House, Folkestone, Kent (Tel. 0303 57421), to whom orders for Volumes 1 to 15 should be sent. Copies of issues for the current year are available from BCS at the above address.

Advertisement office 2 Breams Buildings, London EC4A 1HH Telephone 01-242 3820, 4620, 6320.

The Computer Bulletin is printed by Eyre & Spottiswoode Ltd, Thanet Press, Margate, Kent.

The British Computer Society

Registered Office 29 Portland Place, London W1N 4AP Tel: 01-637 0471
President and Chairman of Council A. S. Douglas BSc, MA, PhD, FBCS, *Deputy President* G. J. Morris ARCS, BSc, DIC, FBCS, *Honorary Treasurer* R. L. Barrington FCA, FCWA, FBCS, *Secretary-General* M. C. Ashill FCA.
A list of the Society's council members appears on page 315. *BCS Library* City University, St John Street, London EC1 Telephone 01-253 1961. 9 am – 9 pm Monday to Thursday, – 8 pm Friday.

Wouldn't it be nice if we could write computer programs in ordinary English—or would it?

I. D. Hill

Laurence Sterne's definition of 'a nose' was as follows [1]: 'by the word *Nose*, throughout all this long chapter of noses, and in every other part of my work, where the word *Nose* occurs—I declare, by that word I mean a nose, and nothing more, or less.'

My definition of 'ordinary English' is precisely similar. Thus, for present purposes I ignore M. P. Barnett's book *Computer Programming in English* [2], for I consider it misnamed; it is really about 'Computer Programming in Snap'. Snap is a language mainly for text-handling that does have a much more English appearance than many programming languages; but it is still a specialised programming language, and not ordinary English by any stretch of the imagination.

I had wondered whether to leave my answer to the question in the title to the end but I think it would be preferable to give it straight away, and then seek to justify it. This is partly because the second part of the title itself makes it pretty clear that my answer is 'no, it would not', but also because I am already in print as saying something of the sort.

Two years ago, I reviewed a book called *Programming Languages* by Jean Sammet [3, 4], and I ended the review with the words

> 'Looking to the future, in Miss Sammet's Utopia we shall be able to talk to computers in English just as we do to people. In my own Utopia, however, we shall be able to write instructions for people in programming languages, just as we do for computers.'

At least one person misunderstood this to mean that I should like to be some sort of dictator, and go around giving people instructions, but in fact this is not the case at all. What I did mean was that there are circumstances where instructions have to be given, either to people or to machines, and that when this is the case, ordinary English is really far too woolly and ambiguous to do the job well. The manner of giving them should be formalised, for otherwise there is far too much risk of misunderstanding.

Take an example given in Miss Sammet's book; a book of which, in general, I have a high opinion. She mentions the program 'calculate the square root of the prime numbers from 3 to 97 and print in three columns'. Suppose we try to execute that program. The first difficulty is what is meant by 'the square root of the prime numbers'? Since 'square root' is in the singular, it is not immediately obvious—it takes a considerable amount of human intelligence to realise that this must mean 'the square root of each prime number' rather than 'the square root of the product of all the prime numbers' for example. Obviously, an interpreter could be written that would take care of this particular point, but that is not sufficient. If we are to be allowed to throw instructions like this around, the interpreter must be able to sort out *any* sort of difficulty that may come up, and that would call for a program to implement *general* intelligence. Can anyone, today, even begin to think how to write it?

Suppose we have got over the first difficulty. Now we have to 'print in three columns', and even using my human general intelligence, I must conclude that I do not know what to print in three columns. I suppose one column must list the prime numbers, and a second one list their square roots. Whatever is in the third one?

Corresponding with Miss Sammet about this example, she has written 'I quite agree that the statement as written is ambiguous; my basic principle is that the user should write his statements and the computer should be in a position to ask for clarification if it is given incorrect or confusing or ambiguous instructions'.

But 'The trouble with people is not that they don't know but that they know so much that ain't so' [5]. And that can be the trouble with computers too. When the computer knows that it does not understand, it can ask for clarification, but what happens when the computer does not ask for clarification, for the instructions are logically clear and unambiguous, but have a meaning totally different from what you thought they meant?

Perhaps it may be argued that this happens nowadays with existing programming languages. Certainly it does, but I suggest that the scope for it is far more extensive when ordinary English is used.

Ambiguities in English

English abounds in examples of ambiguity such as 'list the patients in Ward 10, and tell me what they are in for'.

There are ambiguities of wording like 'You would scarcely recognise little Johnny now. He has grown another foot'.

Precisely similar constructions can mean entirely different things—compare 'call me Edward' with 'call me a taxi'!

Compare 'Do you usually enjoy your holidays?' with 'Do you usually enjoy good health?' A life insurance company would not be pleased if you answered 'Yes' to the latter, meaning 'On the rare occasions when I experience good health, I find it enjoyable'.

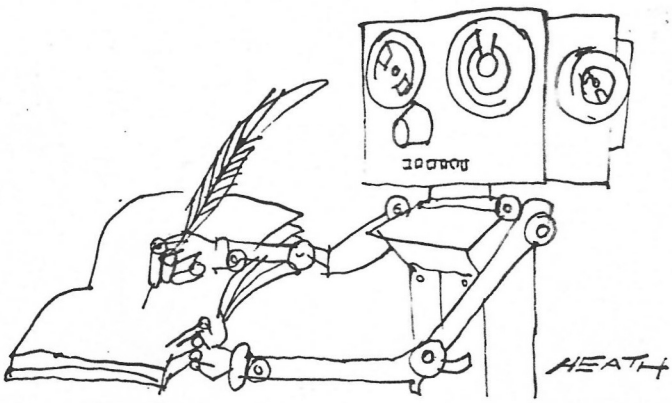
There are the 'Lewis Carroll' type jokes involving words like 'nothing' and 'nobody'. Doubtless, Lewis Carroll would have enjoyed the advertisements that say 'Nothing acts faster than Anadin' and he would surely have pointed out that nothing is cheaper as well as acting faster.

A BBC sports commentator, a while ago, produced the memorable sentence 'The best thing about this match was the shooting of Peter Lorimer'.

Consider the ambiguity of a notice put up in an office building a few years ago: 'During the present fuel shortage, please take advantage of your secretary between the hours of 12 and 2'.

Double negatives

One particularly common ambiguity in English is the double negative, such as 'I don't know nothing' the logical meaning of which is the opposite of its intended meaning. The intended meaning is clear even when the logical meaning is



almost impossible to ascertain, as in 'I don't suppose you don't know nobody who hasn't got no pigs they don't want to sell, do you?'

On the other hand double negatives are quite often used in their literal sense to mean a genuine positive, and an algorithm to sort out which sort of double negative is being used on any particular occasion would be hard indeed to devise.

A recent letter in *The Times* referred to entrants to the Colonial Service being taught that their job would be 'not to "govern others", but to help the people whose servant they would become learn the difficult job of governing themselves'. The writer stated that 'I never met a member of the Colonial Service . . . who gave me cause to doubt that his outlook . . . was different from that built into me', which is ***not***, I am sure, what he intended to say.

Problems of scope

Little words like 'if', 'again', 'repeat' and so on, are notably deficient in English compared with computer languages, in that they do not specify exactly which words are governed by them. There was a joke a few years ago that ran

'I feel like going to bed with Brigitte Bardot again tonight.'

'Again?'

'Yes, I have felt like it before'.

You may say that is only a joke, of no real consequence, but how about this, from the 1968 Finance Act: 'the total deduction from tax . . . shall be reduced, for each allowance if more than one, by an amount equal to tax at the standard rate on £36'? This relates to the 'claw-back' device, whereby family allowances were increased but an adjustment to tax allowances made sure that standard-rate taxpayers had the increase taken away again. A taxpayer argued, and the law-courts eventually agreed with him, that the 'if more than one' governed the entire thing, and that whereas claw-back applied to those with three or more children (i.e. two or more family allowances, since no allowance is given for a first child), it did not apply at all to those with two children and one allowance [6].

I understand that an amendment, with retrospective effect, was put in the 1971 Finance Act, but it seems to me that the lawyers, and their kin—the parliamentary draftsmen—have a lot to learn from computer languages in such matters as the scope of an 'if'.

This scope problem arises in many other places where instructions are given in plain English. Look at the baptism service in the Book of Common Prayer [7].

the priest . . . shall say,

'Hath this child been already baptised, or no?'

If they answer, 'No': Then shall the Priest proceed as followeth.

The 'if' covers the whole of the rest of the service, but it takes common-sense, and knowledge of context to decide that that must be so. One feels that perhaps there should at least be

else the priest shall say:

'What the blazes do you think you are up to then?'

Another scope problem I ran into recently was at a small restaurant where the menu was of the following form

SOUP

.....

COLD MEATS

.....

.....

OMELETTES

.....

.....

MAIN DISHES

.....

.....

Chips and peas included with all the above.

I suppose I should have realised that there was a scope problem about 'all the above' as chips and peas were unlikely to be served with soup, but I was definitely taken by surprise in being asked to pay extra for chips and peas with my cheese omelette; 'all the above' apparently means 'all main dishes'.

Consider also the Pullman Ticket, issued to me for a trip on the Manchester Pullman, saying

FROM		London (Euston)
TO		Liverpool (Lime Street)
	OR	Manchester (Piccadilly)
	OR	Vice Versa

Is there any hope that a computer program could pick up the complete difference in meaning between those two 'OR's?

Examples of programs

Some of the things I have mentioned, while showing ambiguities in English, may seem to have little to do with ideas of programming, although programs in English would be bound to meet such difficulties. But here are some examples that are definitely programs in the computing sense, even though not intended for computers. The first comes from the instructions for putting together the pieces to make a model aeroplane.

The required undercarriage position must now be selected.

13 For a model with retracted undercarriage the main wheels and legs are omitted and the doors cemented in place flush with the underside of the wing.

14 For a model with lowered undercarriage, cement the main wheels onto the undercarriage legs and cement legs into locating bushes inside the wheel wells. Note that the undercarriage legs are angled forward.

15 Next cement the central wheel doors to the under-carriage legs, the pins outside each leg fitting the holes inside each door. Note that the holes on these doors are offset so that the bottom of the door is parallel to the ground.

16 Cement the tab at the top of each outer door into the small cut out at the extreme outboard end of the wheel wells, the door hanging vertically down; cement the lower ends on to the outside of the central doors.

17 Locate and cement the inner wheel doors in place, the small tabs on the end door engaging in the extreme

inboard end of the wheel well, the doors hanging vertically down.

18 Assemble the main gun pack by cementing the ammunition box onto the raised ribs of the flat decking, and cement the deck into the curved bottom panel.

Here we have two alternative actions, for lowered undercarriage or retracted undercarriage, but it is far from clear where the two alternatives join up again. How much better would be the inclusion of some 'go to' instructions to give something like:

The required undercarriage position must now be selected. For a model with lowered undercarriage go to instruction 14.

13 For a model with retracted undercarriage the main wheels and legs are omitted and the doors cemented in place flush with the underside of the wing. Go to instruction 18.

14 Cement the main wheels . . . etc.

That example found difficulties with conditional instructions. Here is one that meets difficulties with looping instructions. Consider the following from a shampoo bottle

For best results wet hair with warm water. Gently work in the first application. Rinse thoroughly and repeat

Repeat from where? Surely the rule must be that, in the absence of other information we repeat from the first instruction. But this means that we have to wet the hair we have just rinsed! Let us use a little common sense and not bother with that. But the next instruction refers to the 'first application' and we cannot do that again, so perhaps logic tells us to miss that one out too. So the only thing left to repeat is 'rinse thoroughly and repeat' and now we are in a closed loop, and must continue rinsing our hair until aborted.

How much clearer it would be to say

```
For best results:
begin
wet hair with warm water;
for j := 1,2 do
  begin
  gently work in application (j); rinse thoroughly
  end
end
```

I do not expect to see anything like that on a shampoo bottle within my lifetime, but I think it is something to be desired, far more than desiring to write plain English for computers.

Figure 1 shows another program with difficulties attached. Let us suppose that the underlining of Congregation and Choir only is sufficient to indicate that these are directives and not words to be sung. Let us suppose that it is clear that 'John Goss' is not to be sung (although the mind boggles at the thought of a computer program that would realise this when suddenly hit by it). Let us suppose that 'Hail! thou ever . . .' is obviously a macro-call to copy the whole chorus. The human mind is capable of deciding these things without much trouble, though goodness knows how!

But we are still in difficulties. First there is a 'bug' in the program—no chorus after the fourth verse. One can feel pretty sure that it is a 'bug', but still sing a nice loud 'Hail' in some trepidation, in case no-one else does.

The real difficulty, however, is where does the Choir only directive finish? Do the choir only sing all the rest (judging from the 'if' in the baptism service, perhaps it ought to mean that)? or all four remaining verses while the congregation join in the choruses? or just the one verse; and if so is it choir only for that verse's chorus also?

I can only report that the congregation, in practice, were far from unanimous in executing this program.

Congregation.	See, amid the winter's snow,	John Goss
	Born for us on earth below;	
	See the tender Lamb appears,	
	Promised from eternal years:	
	Hail! thou ever-blessed morn!	
	Hail! redemption's happy dawn!	
	Sing through all Jerusalem,	
	Christ is born in Bethlehem.	
	Lo, within a manger lies	
	He who built the starry skies;	
	He, who throned in height sublime	
	Sits amid the cherubim:	
	Hail! thou ever . . .	
Choir only.	Say, ye holy shepherds, say	
	What your joyful news today;	
	Wherefore have ye left your sheep	
	On the lonely mountain steep?	
	Hail! thou ever . . .	
	As we watched at dead of night,	
	Lo, we saw a wondrous light;	
	Angels singing "Peace on earth",	
	Told us of the Saviour's birth:	
	Sacred infant, all divine,	
	What a tender love was thine,	
	Thus to come from highest bliss	
	Down to such a world as this:	
	Hail! thou ever . . .	
	Teach, O teach us, Holy Child,	
	By thy face so meek and mild,	
	Teach us to resemble thee,	
	In thy sweet humility.	
	Hail! thou ever . . .	

Figure 1

In addition to these lengthy examples, we have short, direct, simple little instructions such as one to motor-cyclists saying 'Crash helmets must be worn' which seems straightforward enough, until we see the notice by the Underground escalator saying 'Dogs must be carried'.

'Passengers must not cross the line except by the bridge or subway provided' seems reasonable, but what happens when we visit another railway station and find 'Passengers must not cross the line'? Are we stuck on one side of the line for life?

Regulations on Brighton seafront are reported as banning the use of a vehicle for 'cooking food or heating liquid'. Does that include radiator water?

But then, if we start looking at regulations involving vehicles, we find in the Highway Code [8]:

Before driving, *make sure that*

. . . . . .

—your speedometer is in working order;

Can anyone suggest how?

Punctuation

Punctuation is a difficulty that people sometimes complain about in computer languages—they left out a semi-colon, or put in a comma, with troublesome consequences. Ordinary English does not overcome this for us. Indeed, it is worse, in that mispunctuation in Fortran or Algol will usually lead merely to an error message, whereas mispunctuation in English can change the meaning.

Compare 'British Railways hope to have trains running normally, late this afternoon' with what a London evening

paper printed: 'British Railways hope to have trains running normally late, this afternoon'.

Compare 'The Prime Ministers called for an end to violence and internment, as soon as possible' with what the BBC newsreader seemed to read out: 'The Prime Ministers called for an end to violence, and internment as soon as possible'.

And how about the invitation in the Communion Service 'Drink ye all of this'? Does it mean 'Drink ye all, of this', or 'Drink ye, all of this'?

There are difficulties with notation. Computer languages are at least consistent in their use of mathematical signs, but English uses / and - with great abandon. The tax year 1971/72 can also be written 1971-72 without either division or subtraction being meant. A minus sign can even be used to mean division, in a financial context where a price of 3 9-64 means $3\frac{9}{64}$. Notice also how space dependent this is: 3 9-64 is very different from 39-64.

If you have an interpreter that has allowed for these usages, can it be expected not to hiccup when an advertisement in *The Times* says

'CUP FINAL TICKET, 1/2 required'

I should love to see some automated system sending off ½ a ticket.

Interpreting numbers

There are also difficulties with numbers. 'Twelve and a half' means 12·5, and therefore we should expect 'A million and a half' to mean 1,000,000·5, but usually it means 1,500,000. If we allow for that, there are two difficulties. Firstly, what happens to 'a hundred and a half'? Would that mean 150? Secondly, how does one pronounce '1,000,000½' when one actually means that?

Arithmetic of percentages is appallingly lax in plain English. An increase in Bank Rate from 5 per cent to 6 per cent is usually called a 1 per cent increase, but the rate has been increased by 20 per cent of its prior value.

And while dealing with numbers, imagine a computer program to deal correctly with the following recommendation for how much doctors should be paid [9]

> a basic practice allowance of £1000 a year, with a standard capitation fee of 1s. a year for each patient under 65 and 1s. 8d. a year for patients over 65. For each patient over 1000 there would be a supplementary capitation fee of 2s. 6d. a year.

Quite apart from the difficulty of 'over 65' meaning age, while 'over 1000' means number of patients, we also have the less obvious but equally disastrous contrast between '*each patient* under 65' and '*patients* over 65'. Presumably only one amount of 1s. 8d. is paid for all the over 65 patients lumped together. We could also run into trouble with those equal to 65—the program would have to know that with ages, $>$ and $\geqslant$ seem to have identical meanings.

Positive and negative

In the medical field we meet the situation where a technician looking down a microscope at a blood-sample, or some such, counts how many of a particular feature he can find in a given area. If the answer is zero, he is said to have a 'negative' specimen.

Suppose this process were automated, with a program saying 'Count the number of whatnots found, and report whether the result is negative'. One can see the difficulties that would arise from forgetting to mention that 'negative' means 'zero' in this context.

In the context of breath- or blood-testing for alcoholic content, a 'negative result' seems to mean any value less than 80 mgm/100 ml (i.e. the level at which driving a car becomes an offence).

Technical terms

How plain is plain English? What is meaningful to one person is complete gibberish to another, and obviously any interpreter would have to understand technical terms relevant to the context. But having taught a computer technical terms, how would you avoid the technical meaning being used in a context where a non-technical meaning was intended? Words like 'significant' in statistics would be very dangerous indeed.

The Devon and Cornwall Police recently put out a message which said

> *Quit scuffing your creepies, man. Do it like now.*
>
> Grip this. Some yhuk with a perch for boo boards has dipped plenty on this scene. If you're not formatting with the weepies on a loss awareness of your boo board, nix out on the fade with it stashed on the moke or cooling on the salt grip.
>
> Spread by the fuzz of Devon and Cornwall to help sock it to the mean cats.

Is that plain English? Obviously not, but how about this?

> For the purposes of the pool betting duty, any payment which entitles a person to make a bet by way of pool betting shall if he makes the bet, be treated as stake money on the bet, and this subsection shall apply to any payment entitling a person to take part in a transaction which is, on his part, only not a bet made by way of pool betting by reason of his not in fact making any stake as if the transaction were such a bet, and the transaction shall accordingly be treated as a bet for the purposes of the pool betting duty.

As Lord Brabazon remarked 'Whoever drafted that must have had something in his mind—God knows what'. Well, that is an attempt to write a precise and unambiguous program in English.

That legal language is so 'heavy' should act as a warning that writing unambiguous English is difficult.

Sometimes lack of knowledge of technical terms can give not merely lack of information, but utterly misleading information —consider the newspaper headline reading 'Lady Margaret bumped by Jesus'.*

English arithmetic

Even the English way of expressing arithmetical operations is not always as definite as one could wish. A boy was once asked in an examination question 'How many times can you take 6 away from a million?' He replied

1000000	1000000	1000000
6	6	6
999994	999994	999994

I can do this as many times as you like.

I hope he got full marks; he had answered the question that was asked.

And, of course, plain English has little in the way of algebraic notation. We should surely have to incorporate ordinary algebra, as well as ordinary English.

To be fair to Miss Sammet, whose book I mentioned earlier, I must report that, in favouring natural language, she says 'this concept *definitely includes* mathematical or any other relevant scientific notation' [her italics].

*'Lady Margaret' and 'Jesus' are boats in the Cambridge University bumping races.

'Theirs not to reason why'

People sometimes complain when computers do literally what was in the program, however stupid it may be, but it does not take a computer to do this. Put humans in a situation where they have to follow instructions precisely and equally ludicrous results can arise.

A few years ago, on a Saturday, a man was fined £1 for being drunk and disorderly, refused to pay and was sent to prison for seven days instead. Under prison regulations he was searched and found to have £1.5s. in his pocket. Again under regulations the pound was taken to pay the fine, and he was discharged, after 15 minutes in prison.

But the regulations also said that anyone released from prison on a Saturday had to be given £4.10s. to tide him over the weekend. So he made £3.10s. profit altogether. And no computer was involved in that story.

Another interesting case of a program going wrong, without a computer being involved, concerned a certain Mr Gutteridge who was waiting to give evidence in a motoring case. He was asked if he was Mr Gutteridge, and he was. He was asked to come in, so he did. He was asked to read from a printed form, so he did. He was then congratulated on having become a special constable, while another Mr Gutteridge was still waiting outside to be called.

At least when a computer follows a ludicrous program, it is not itself (so far as we know) aware of what it is doing. On the other hand, humans can be persuaded to follow programs when they should know better. There is, for example, the 'Quit Rent Ceremony' performed before the Queen's Remembrancer in the High Court of Justice, a report of part of which reads

> The Chief Clerk called upon the tenants and occupiers of a piece of waste ground called 'The Moors' in the county of Salop to come forth and do service.
>
> The Comptroller and City Solicitor attempted several times unsuccessfully to cut a fagot of hazel twigs of one year's growth with a blunt billhook, but severed another with a sharp hatchet, whereupon, when the billhook and hatchet were tendered, the Queen's Remembrancer said 'Good Service'.
>
> Then were called forth the tenants and occupiers of a certain tenement called 'The Forge' in the parish of St. Clement Danes in the County of Middlesex to do their service by counting six horseshoes and 61 nails, whereupon the Queen's Remembrancer said 'Good number' on each occasion.

Computer training

Perhaps with some of the preceding examples one might say that such a result could only come from someone completely untrained in any aspects of computing. With people who know about computing we should get better results. Yet look at the NCC Manual on Standard Fortran [10] (on the whole an excellent publication) to find instructions such as

> Please read the introduction to the manual first.
>
> The manual is divided into four parts. Please read the introduction to each of the parts before you use that part.
>
> .....
>
> Please read all four parts of the manual (after you have read the introduction) before you start writing Standard Fortran programs. (Probably it is best to start with Part II first of all).

Does 'first of all' over-ride the 'Introduction . . . first'?

The same manual had a list of amendments, including

> 'Page 133, line 12; for (2) read (1)'

where Page 133, line 12, read

> (2) Actual argument is an array name (see note (2)).

Only the (2) at the end of the line was meant to be changed.

Another amendment in the same list asked you to add the word 'nest' to a sentence, and the following amendment then told you to delete the entire sentence and replace it by something else.

The idea that people with computer training might do better takes a severe blow from an incident at a 1968 meeting of IFIP's Working Group on Algol.

This was attended by 26 people, all Algol experts, plus a number of observers, including myself. One afternoon they had an exercise in which a number of topics for future discussion were proposed, and each of the experts was asked to copy the list, and that evening to put not more than 10 x's against those items he thought most important.

'Most important for this committee, or for the world in general?' was asked, and it was agreed to have two columns, one for each.

When the results came to be analysed next day, the whole thing was found to be totally worthless as the instructions had been interpreted in three different ways by different people, namely

1 **if** important for the world **then** mark (1); **if** important for this committee **then** mark (2);
2 **if** important for the world **then** mark (1) **else if** important for this committee **then** mark (2);
3 **if** important for this committee **then** mark (2) **else if** important for the world **then** mark (1).

Many expert-hours were wasted over this, and some of the experts had come long distances to be present, and so such time wastage was not a trivial matter.

Questions of definition

Another potential source of trouble lies in the difficulties of defining what words mean. Any dictionary, of course, must consist of closed loops since the words used in definitions must all appear in the dictionary. Has any dictionary ever followed the mathematical pattern of first listing words whose meanings are considered axiomatic, and then building all the rest on those foundations?

It is well known that looking a word up in a dictionary, and then replacing the word by any definition found, may be a dangerous habit.

One meaning given for the word 'nothing' is 'a trifle'; but 'there is nothing for lunch' does not mean 'there is a trifle for lunch'.

My son, reading that a certain aeroplane was used for liaison duties, asked me what 'liaison' meant. As a useful exercise, I sent him to look it up in the dictionary, and the first meaning given was 'an illicit sexual adventure'.

The word 'obscene' has recently caused trouble, and a correspondent in *The Times* has said that it should mean 'repulsive, filthy and lewd'. But does it really need to be all three, as the word 'and' would imply?

Suppose something is filthy and lewd without being repulsive? Would it cease to be obscene? If it is argued that this is an impossible combination of attributes, then 'repulsive' is redundant and the definition could be simplified. Similarly, I think that 'filthy' might be found redundant, and we would be left with 'obscene' means 'lewd'—and is 'lewd' any better defined than 'obscene' was before we started?

But then the present law on obscenity apparently requires a tendency 'to deprave and corrupt', whereas one would have thought that 'deprave *or* corrupt' should have been sufficient.

Definitions of words can vary from place to place, not only between English English and American English, but even

within England a word can be differently defined in different parts of the country.

Even a word that actually is used in computer languages can suffer this fate. In Yorkshire the word 'while' means what 'until' means in other parts of the country; when the layout for unmanned level crossings included a notice saying 'Do not cross while red lights are flashing' the danger of this usage became apparent, and new notices had to be produced saying 'when' instead of 'while'.

Cookery programs

Cookery tries to use ordinary English, but is often ambiguous as a result. Does 'Serve hot or cold with cream' mean 'serve hot (or cold with cream)' or 'Serve (hot or cold) with cream'?

Perhaps it does not matter much as it is merely a matter of taste. But that is the sort of English that some people would like to use for computer programs, and surely it would then matter very much indeed.

The use of 'and' and 'or' in English, and particularly expressions involving several 'ands' and 'ors', leaves much to be desired in the way of exact definition of meaning.

How about 'Bake at 350°F for 40 minutes, or until soft'? Does this mean that you stop after 40 minutes even if the apples are still hard? Or does it mean that you must go on until 40 minutes, even if they are soft after 30?

A well-known cookery algorithm states that to make a pot of tea for n people, you use $n + 1$ teaspoonfuls of tea, or as it is often quoted, 'one for each person and one for the pot'. My mother once said that this was all nonsense, and, when my father enquired how much tea should be used then, gave the remarkable reply 'as much as you think you need'.

Surely the whole point of recipes should be to avoid any guesswork. Yet look at these instructions for cooking a turkey:

8-10 lb	325°F	4 -4½ hours
12-16 lb	300°F	4½-5 hours
16-18 lb	300°F	5 -5½ hours
20-24 lb	300°F	5½-6 hours

Suppose the turkey weighs 11 lb, what then? One has to rely on guesswork interpolation. How much simpler it would be to say something like

if the turkey weighs w lb. then cook at

$$(\text{if } w < 12 \text{ then } 325 \text{ else } 300)°F \text{ for } (3 + \frac{w}{8}) \text{ hours}$$

Knitting and music are more interesting than cooking in that, before the computer-age, both those arts found ordinary language inadequate and had to invent a programming notation. (I say 'language' rather than 'English', since Italian is the ordinary language of music).

Knitting

Knitting I know little about, and it has been dealt with by Bryan Higman [11]. I shall therefore add only two remarks: 1) knitting does demonstrate that the general population can manage to understand a programming language, given the motivation; 2) for those who have learned to follow a knitting pattern, what a ridiculous step it would be to try to use ordinary English instead, and to imagine that that was a great advance. (I must note however that I once said that to a lady, who replied 'Oh, but that is ordinary English').

Music

Musical programming is a mixture of Italian words, strange signs, and conventions that every trained musician understands.

Music and computing have more in common than might at first be supposed, for consider the unfortunate pianist who got into a closed loop during the slow movement of Rachmaninov's second concerto at a Promenade concert a few years ago and couldn't find her way out!

On the whole, musical notation works pretty well, so why change it? Yet if real clarity is wanted, I would suggest that the musicians, like the lawyers, could learn a lot from the computer programmers if they were willing to.

As a simple example, let us consider 'minuet and trio' form, as met in the third movement of nearly every Mozart symphony, for instance. Something like that shown in Figure 2.

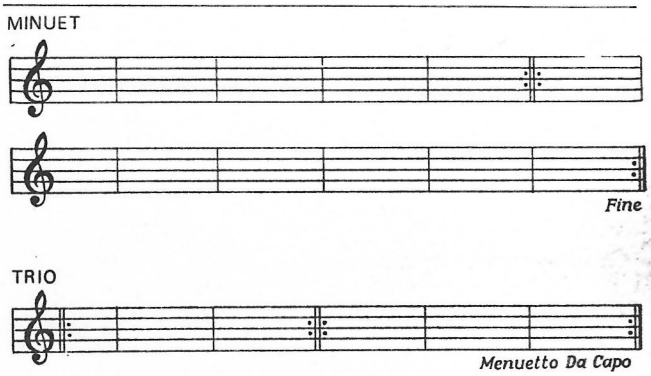


Figure 2

Knowing that 'Fine' means 'End', that 'Menuetto Da Capo' means 'Return to the start of the Minuet', we still need several conventions to tell us that we avoid closed loops by ignoring :|| when we are already repeating, that we ignore 'Fine' the first twice that we get to it, that we ignore 'Menuetto Da Capo' the first time we reach it, and that we ignore all the repeats the final time through the Minuet.

All these conventions could be avoided, and the whole thing made explicit, by writing the *form* of such a movement once and for all as

```
procedure minuet and trio (minuet part 1, minuet part 2,
          trio part 1, trio part 2);
for j := 2, 1 do
  begin
  j times do minuet part 1;
  j times do minuet part 2;
  if j = 2 then
    begin
    twice do trio part 1;
    twice do trio part 2
    end
  end minuet and trio;
```

Any particular such movement could then be written as a subroutine call giving the actual parameters as shown in Figure 3.

The difficulties with relying on conventions are

1) what happens if the conventions are forgotten? This happened with pre-classical music, where the notation was

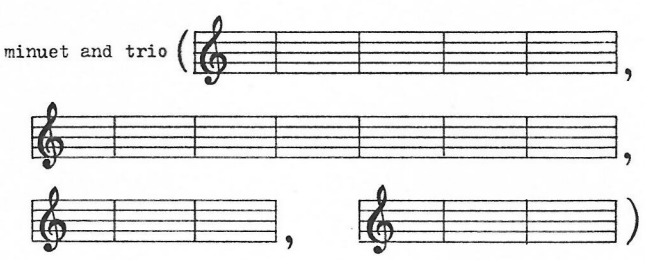


Figure 3

intended merely as an aid to memory rather than a complete program, and nowadays we cannot reconstruct it with certainty. Even later, in Mozart piano concertos for instance, there is argument over whether the soloist should play what Mozart wrote, or follow the convention of Mozart's day when soloists would be expected to decorate their part;

2) what happens when you want to break the convention? In Beethoven's day, every educated musician knew that, in the first movement of a concerto, when the orchestra paused on a loud dominant chord the soloist was expected to continue by improvising a cadenza. In the Emperor Concerto, he had to write (in Italian) 'Do not play a cadenza here, but go straight on with what is written' as the only way to break the convention.

But at least music has recognised that, where instructions are to be given, a precise notation is needed.

Talking to computers

Some people would like to be saved the bother of even having to write down their programs, and to be able to speak directly. There is clearly a case, in some circumstances, for being able to supply *data* by voice.

The dangers of *programming* by voice lie partly in the difficulty of remembering where you have got to when not writing things down, but mainly, I think, in the possibility of being mis-heard.

Sir Alec Douglas-Home, at the 1964 general election, asked for another 'tenure of office' for the Conservative Party, and was criticised by *The Times* for asking for another 'ten years of office'. Imagine a computer mis-hearing something like that, and acting on the mis-information.

I also know of a case where the spoken words 'desk machine' were typed as 'death machine'. Imagine how serious the consequences of that could be.

Even when the words have been heard correctly, it may not always be possible to tell their meaning without further information. When the Archbishop of Canterbury preached in South Africa recently, a BBC news item referred to 'the size of the congregation' . . . or was it 'the sighs of the congregation'? It was impossible to tell from the context.

Conclusion

Suppose that all the difficulties that I have mentioned could be overcome, that someone actually did succeed in writing an interpreter (or even a compiler), for ordinary English programming, that would accept all the strange things it might be given, and make sense of them. Suppose also that he had a large enough computer store to contain the amount of information that would be necessary, for the computer to be able to make sensible judgments about what things meant, by working on the context.

Would things really be any easier? The main difficulty in programming lies in deciding *exactly* what is the right thing to do. To put it into a programming language is relatively trivial. This would simply lead to the idea that any fool could write a program—and we have quite enough rubbishy programs clogging up the works even now.

And would we not have thrown away one of the greatest advantages of computers? Namely that they can be instructed to do exactly what you want without argument or misunderstanding. True we should still have the computer's speed, accuracy and lack of boredom. But as far as getting our instructions understood was concerned, we should be back to the bad old days, when we had to put up with 'biological computers' who said 'Oh! But I thought you meant so-and-so'. This is not Utopia, to my mind.

Some people may agree with me to the extent of saying 'I agree then that ordinary English is not desirable. But could we not have a language, with an exact interpretation, consisting of a formalised mixture of English words, and numerical and algebraic symbols?' If anyone says that, I reply 'Yes. We have already got one. It is called Algol'. But that should be another story, rather than the tail end of this one.

Let me finish as I started, with a reference to Jean Sammet. I do not wish to give the impression that she and I are conducting a private war, but we have had an interesting correspondence. She has written

> 'I have never contended that English itself (or any natural language) is precise. Thus I am not advocating it as a model to be used for clear communication. I am suggesting that in the world as a whole it is desirable to allow people to communicate with a computer the same way that they communicate with each other. Certainly there will be misunderstandings, but I choose to believe that these misunderstandings will be less harmful than the slowdown to progress resulting from forcing all people to learn some higher level language such as ALGOL in order to use a computer.'

I agree that it is desirable that people should be able to approach a computer without having to 'change gear'. But I would start at the other end—not teaching computers English, but teaching people to communicate with each other (*not* in ordinary conversation, but only when giving instructions) in a more precise way.

Could we not start with lawyer's English? This year's Finance Bill [12] has introduced the regulations for Value Added Tax, expressed in lawyer's English, which will now have to be interpreted by computer programmers throughout the country in a great many commercial and financial programs. It is too late to modify the course of events now, but would it not have been preferable, from every point of view, had the Bill contained the necessary regulations in algorithmic form (perhaps as a flow diagram) in the first place?

Acknowledgments

I am grateful to Miss Valerie Coulson for pointing out the ambiguity on the Pullman ticket, and to Dr. C. C. Spicer for introducing me to the 'death machine'. I am very grateful to Miss Jean Sammet for allowing me to quote from our private correspondence.

References

1 Sterne, L. 1781. *The life and opinions of Tristram Shandy, Gentleman*. Book III, Chapter XXXI.

2 Barnett, M. P. 1969. *Computer Programming in English*, Harcourt Brace and World Inc.

3 Hill, I. D. 1970. Review of: "Programming Languages: History and Fundamentals", *The Computer Bulletin*, Volume 14, Number 1, page 14.

4 Sammet, J. E. 1969. *Programming Languages: History and Fundamentals*. Prentice-Hall International.

5 Billings, J. 1874. *Josh Billings' Encyclopedia of Wit and Wisdom*.

6 *The Times* Law Report, 5 March, 1971.

7 *Book of Common Prayer of the Church of England* (1662).

8 *Highway Code* 1968, pages 47-48. Her Majesty's Stationery Office.

9 *The Times*, 5 May, 1966.

10 National Computing Centre. 1970. *Standard Fortran Programming Manual*. NCC Limited.

11 Higman, B. 1967. *A Comparative Study of Programming Languages*. Macdonald.

12 *Finance Bill* 1972, Part I, Her Majesty's Stationery Office.

About the Author

I. D. Hill, B.Sc., studied statistics at University College London. He was employed in the Statistical Advisory Unit of the Ministry of Supply until 1961, when he moved to the Medical Research Council's Statistical Research Unit. In 1967 he transferred to the Council's Computer Unit. He has been a member of the BCS since 1964.