Computer Assisted Learning **Languages** **Graham Davies**

NEW TECHNOLOGIES FOR LINGUISTS

The author describes the first steps in CAL now being taken at the Ealing College of Higher Education, Ealing, London.

Ask the average linguist, particularly the teacher of languages, for his reactions to the microprocessor revolution, and you are likely to meet with an indifferent or negative response. Frequently this attitude stems from suspicion or fear, a refusal to accept that the ubiquitous chip is already well-established in the linguist's sphere of activity and shows all the signs of staying there. But this suspicion or fear is often unfounded and based only on ignorance.

The truth of the matter is that linguists can no longer afford to be ignorant about computing. Hand-held, battery-driven, phraseological translators are already being marketed. Computer dictionaries, accessed via a VDU linked to a mainframe, are well-established in the EEC and many government and commercial organisations. In Luxembourg, work on SYSTRAN, an automatic translation package, is already at the advanced stage. The computer can sort words alphabetically, thematically, in reverse order or any way you like. It can search through lists in seconds. It can process texts, tell you if Bacon wrote Shakespeare, and print out your work faster than the fastest typist – all neatly spaced and centrally aligned. In short, it crunches and digests information at an alarming rate, saving time, energy and money. And let's face it, it is the information business we linguists are in.

The most compelling reason for taking note of the sweeping changes being brought about by the microprocessor is that our inertia will inevitably lead to those with expertise in computing calling the tune. Software development (writing the material which makes computers perform their weird and wonderful tasks) is too important a job to be left to the programmers. Contrary to popular opinion, programmers do not have to be mathematical wizards, although some skill in arithmetic and algebra helps; they merely have to be clear-headed, meticulous and logical. Linguists often fit into this category – in fact the National Computing Centre has discovered a strong link between success in programming and linguistic ability, in one's mother tongue in particular. If we wish to hold our own in the world of the future, we must either become efficient programmers ourselves or at the very least know what we should tell those who are. Programmers do not necessarily understand specialist requirements, and the result is often sloppily-written and inadequate software.

Over the last two years, a small group of linguists in Ealing College have been trying to catch up on recent developments. Using the college's Prime 300 computer, we have gained reasonable proficiency in programming in BASIC, an easy to learn programming language. Two of us, David Steel and myself, attended a course at the University College of Wales in Aberystwyth and learned how to program in SNOBOL4 and use a concordance package (COCOA). We have also picked up an elementary knowledge of FORTRAN IV.

Since our involvement in computing we have created on the Prime 300 a demonstration multilingual dictionary of economics and politics, a language synthesiser (which writes 'poetry' according to rigid syntactic rules but random formal and semantic selection processes), and a number of concordance, word-frequency count and lexical analysis programs. However, our main area of research at present is into Computer Assisted Learning, or CAL. I am especially grateful for advice given by Dr. Rex Last of the University of Hull, who sparked off our interest in this subject.

Machines as aids to teaching languages are something with which linguists are already familiar. The advent of the language laboratory in the 1960s (when Ealing took the lead) caused a minor revolution in language teaching. Suddenly the drudgery of drilling students in oral/aural skills was taken over by technology. However, the language laboratory soon revealed its shortcomings. Students had to be constantly monitored, especially beginners. The dulcet whispers in the headphones proved to be more effective than counting sheep; tapes snapped violently, got into awesome tangles; students verged on nervous breakdowns as the disembodied voice rambled on inexorably. Finally, the language laboratory did nothing to monitor writing skills.

Compared with the language laboratory, the computer is infinitely more flexible. Providing the teacher (or programmer) knows what he wants to do, it will do it – precisely and quickly. CAL is particularly relevant to language teaching, especially in the early stages, as there are numerous mechanical aspects, eg. conjugations of verbs, agreement of adjectives, cases, systems of pronouns etc, which learners find difficult and which teachers find tedious to constantly check and drill. So why not let the computer take over the checking and drilling in these areas and allow the teacher to concentrate on more interesting things, such as expression, style, essay writing and conversation? It is with this aim in mind that we began to put together our CAL package, which, incidentally, is flexible enough to be used in *any* discipline where there is a need for testing knowledge of facts, eg. history, geography etc. The following description shows how the package works in practice:

> Since this article was written, Ealing College has been successful in obtaining a generous grant from BP Oil to further our research. Our first step was to obtain a multi-user Tandberg EC10 microcomputer. Now we intend to adapt many of our programs for use on this machine, and with the additional facilities we propose making the teaching of German by computer an integral element of one of our courses.

Getting on line

First, the learner has to get on line, which begins with a search for a free terminal, either a VDU (Visual Display Unit – like a TV set) or a hard-copy printer (a teletypewriter). The terminal is linked directly to the computer, in which the CAL package is stored, and essentially it is little different from a typewriter. If a hard-copy printer is used, first it must be switched on. VDUs are usually left switched on. The student then types LOGIN ALS1 (plus the RETURN key – sometimes marked CR). The RETURN key relays the typed message back to the computer and must be pressed whenever information is sent from the keyboard to the computer. LOGIN ALS1 should result in the computer replying ALS1 LOGGED IN, with an indication of the time etc. If the student makes a typing error he can erase individual letters by typing " or cancel longer messges by typing ?. Thus ALL"S1 is interpreted by the computer as ALS1. ALLL""S1 is also read as ALS1. ADHDHFHF?ALS1 is read as ALS1. Note that these so-called 'control' characters work retroactively, the question mark going back to the beginning of what the student typed in and the double quotation mark going back one letter at a time. Once he gets a LOGGED IN message back from the computer, the student simply types LBASIC (plus RETURN). This tells the computer he wishes to communicate with it in BASIC. The computer replies with:

```
GO
>
```

Now the student can load the program he wishes to use.

GDTEST – a flexible testing program

Our main testing program is called GDTEST. To run this program the student first types:

LOAD 'GDTEST' (plus RETURN key)

The single quotation marks indicate the program name here. On receiving this instruction, the computer should reply with:

```
>
```

when it is ready. Now the student types RUN (plus RETURN), and the program is executed.

What follows is a series of requests for information and then the test itself. Some requests may require a YES or NO answer, which may be abbreviated to Y or N. If the student types MAYBE, where YES or NO is expected, he is told that the computer does not understand his message and given another chance to reply.

Essentially, the session at the keyboard takes the form of a dialogue between student and machine, the computer doling out praise where due, commiserating when wrong answers are given and offering help when the student gets into difficulties. This is a major step forward from the language laboratory session, and gives the student the impression the machine understands his problems. The necessity of acquiring motor skills, ie. reasonable typing proficiency, is an advantage rather than a handicap, in that the student has to use his hands constantly and remain alert throughout the testing session. Ideally, however, student responses should be as short as possible so as to avoid input errors. The control characters, already mentioned, can of course be used when replying to questions.

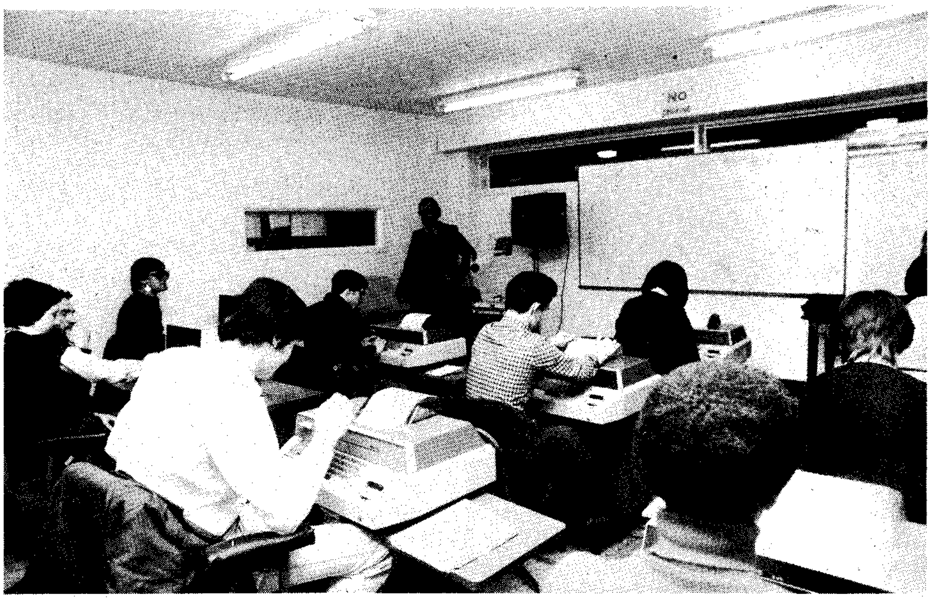
GDTEST reads from a selection of tests stored in data files. Each test has a header containing explanatory remarks and examples. Tests developed so far assume that the student has already been taught the points being tested, but there is no reason why the header should not contain new material (providing it is not too complex), which might then be reinforced in the subsequent question-answer routine. Additionally, the header may give the student information regarding other tests in a series.

A test may consist of up to 50 questions, but in practice 20-30 is usually enough to cope with. Any sort of question is possible: gap-filling exercises, those requiring single word or whole sentence answers, questions to which there is more than one possible answer, questions in any language. Each question may occupy up to 4 lines of typescripts (note a line is 72 characters long, including blanks, on most terminals). These 4 lines could be expanded to, say, 5 or 6, but 4 seems sufficient for the time being. Within these 4 lines there may be explanatory comments, eg. 'Note the feminine singular is required in this question' or 'The next 5 questions require you to convert nouns into pronouns'. The answer area in the data file allows for a maximum of 4 lines of typescript. These 4 lines may be used in different ways: each line may contain one of a maximum of 4 alternative answers or various comments to follow correct or incorrect student responses to questions (see below). These 4 lines seem sufficient at present, but expansion is easy, if required.

The program breaks up the test into blocks of any size the teacher chooses (10 is often convenient, but any figure is possible, providing it remains constant for the whole of any single test) and each question, including comments if attached, is presented to the student, who then types in his answer at the keyboard (not forgetting the RETURN key). If he does not know the answer he simply types 'X'.

If he is right he is told so, and may be congratulated. Congratulatory comments, or reinforcement comments conveying additional information for the student, which follow correct student responses, are stored in the answer area in the data file and distinguished from answers by their prefix ' . . . ', eg. ' . . . Good, that was a hard one!' or ' . . . Note that fahren is also used transitively – Herr Fuchs faehrt Auto'.

If the student gives a wrong answer he is immediately told what the correct answer should be (the first, where alternatives are stored). At this point a commiseratory comment may appear or an explanation as to how the correct answer is arrived at. Comments to follow incorrect responses are prefixed by a dash, eg. '– Bad luck, see the section on the genitive in the notes to this test'.



At the end of each block the student is given his score. If he answers more than 3 questions incorrectly he is automatically offered notes to assist him, which he can accept or refuse. These notes may also be summoned at any stage during the test by typing 'H' (ie. HELP!) instead of an answer. After the help notes have been presented, or if the student refuses the offer, a remedial rerun is immediately initiated, whereby any unattempted or incorrectly answered questions are presented again. Any comments accompanying questions which are not required to be repeated on reruns (eg. 'Note the next 5 questions refer to feminine adjective forms') may be suppressed by prefixing the comment line in the data file with a dash (–). Remedial reruns continue until the student scores full marks. This is to encourage the learner not to leave the keyboard until he has given at least one correct answer to each question.

The process may be compared with what frequently happens in the language laboratory, where the student may plough gaily through a dozen drills, understand none of them and get them all wrong – unless of course the teacher is at hand to cut in. The computer never tires of monitoring and correcting.

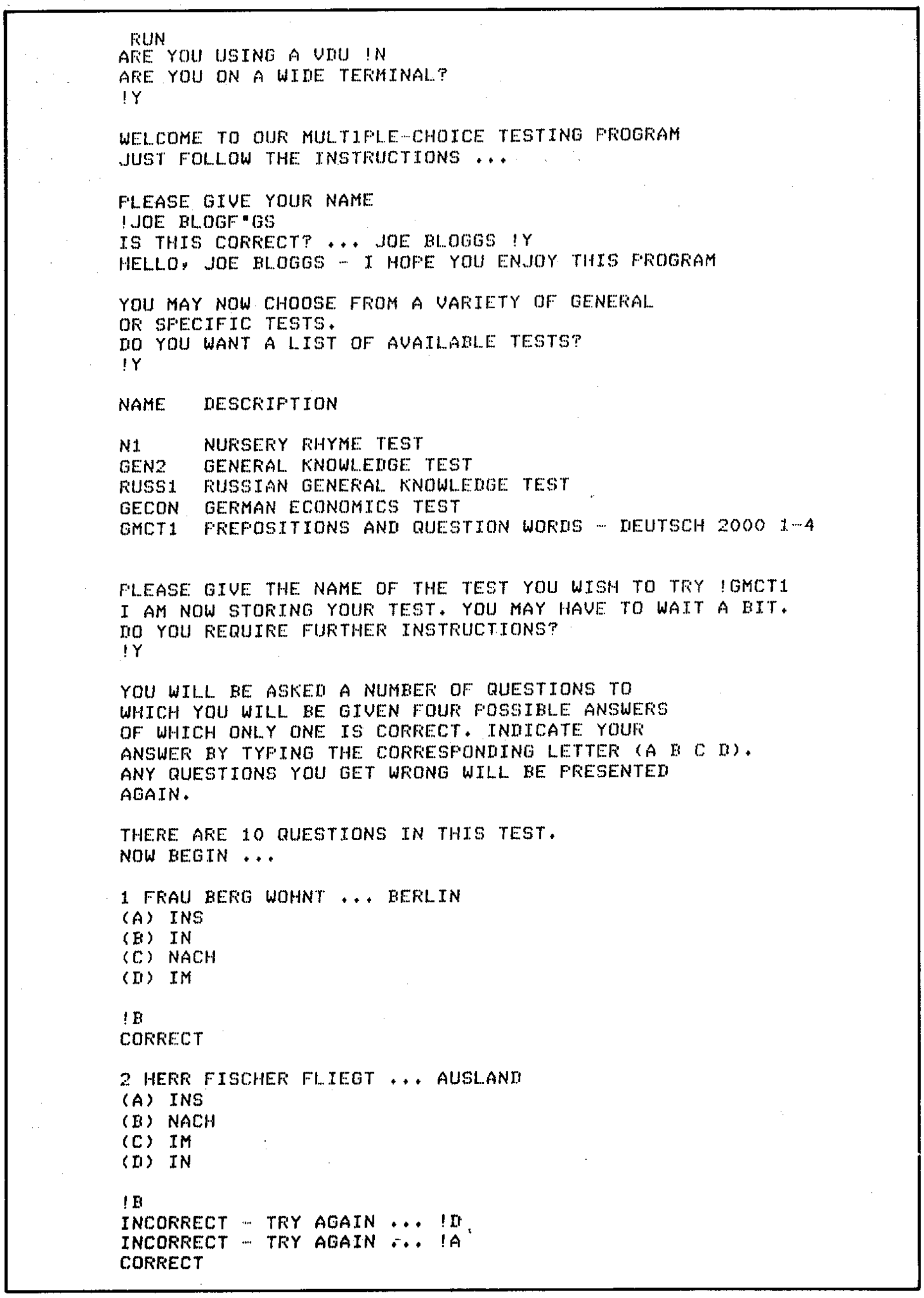
Having completed a block, the student may go on to try another if he wishes, and so on until the test is completed. Should the student get completely bogged down, or realise that he has clearly chosen an unsuitable test or is already 10 minutes late for a lecture, he can type 'SOS' in reply to any test question. This opens the escape hatch and lifts him safely out of the current block. He can then refuse further questions and quit the test program. Too brutal tactics could prove very irritating, so this facility at least leaves the learner's options open.

If a student continually gets questions wrong he is encouraged to keep trying but is also reminded that he can quit the block by using the SOS facility. If he perseveres, help notes are then automatically presented and a remedial rerun is initiated, going over what he has covered up to that point. This prevents reruns becoming too long. Once all the blocks in a test have been completed, a new test is offered, which the student can accept or refuse. Ideally, we would like to build in some sort of intelligent branching procedure at this stage, whereby the computer offers progressively more difficult tests or, in the case of the weaker student, a series of back-up or remedial tests. This is all within the capability of the computer, but requires much more research.

When a student finishes his session on GDTEST, he is given his score, expressed as the percentage of correct answers out of all questions presented. This is nearly always sufficiently high to encourage him to come back for more.

During the course of each test the computer makes a record of the student's performance, storing details of which tests were attempted, how many attempts were made at each question, whether the test was abandoned before completion etc. A program has already been written to read from these records and produce individual student profiles, which may, if desired, be handed to the students concerned. Work is also in progress on programs to produce group profiles and test profiles. This should aid error analysis and reveal shortcomings in badly-designed tests.

At present files are created in the EDITOR, which can be a bit tricky for the inexperienced user, so beginners are advised to seek help from someone familiar with the EDITOR. Tuition can of course be provided by our team, subject to time and demand. Work is in progress on a program designed to facilitate test creation and make the whole process idiot-proof. This will be released once we are satisfied that we have a test-structure to suit all tastes.



```
 RUN
ARE YOU USING A VDU !N
ARE YOU ON A WIDE TERMINAL?
!Y

WELCOME TO OUR MULTIPLE-CHOICE TESTING PROGRAM
JUST FOLLOW THE INSTRUCTIONS ...

PLEASE GIVE YOUR NAME
!JOE BLOGF*GS
IS THIS CORRECT? ... JOE BLOGGS !Y
HELLO, JOE BLOGGS - I HOPE YOU ENJOY THIS PROGRAM

YOU MAY NOW CHOOSE FROM A VARIETY OF GENERAL
OR SPECIFIC TESTS.
DO YOU WANT A LIST OF AVAILABLE TESTS?
!Y

NAME   DESCRIPTION

N1     NURSERY RHYME TEST
GEN2   GENERAL KNOWLEDGE TEST
RUSS1  RUSSIAN GENERAL KNOWLEDGE TEST
GECON  GERMAN ECONOMICS TEST
GMCT1  PREPOSITIONS AND QUESTION WORDS - DEUTSCH 2000 1-4


PLEASE GIVE THE NAME OF THE TEST YOU WISH TO TRY !GMCT1
I AM NOW STORING YOUR TEST. YOU MAY HAVE TO WAIT A BIT.
DO YOU REQUIRE FURTHER INSTRUCTIONS?
!Y

YOU WILL BE ASKED A NUMBER OF QUESTIONS TO
WHICH YOU WILL BE GIVEN FOUR POSSIBLE ANSWERS
OF WHICH ONLY ONE IS CORRECT. INDICATE YOUR
ANSWER BY TYPING THE CORRESPONDING LETTER (A B C D).
ANY QUESTIONS YOU GET WRONG WILL BE PRESENTED
AGAIN.

THERE ARE 10 QUESTIONS IN THIS TEST.
NOW BEGIN ...

1 FRAU BERG WOHNT ... BERLIN
(A) INS
(B) IN
(C) NACH
(D) IM

!B
CORRECT

2 HERR FISCHER FLIEGT ... AUSLAND
(A) INS
(B) NACH
(C) IM
(D) IN

!B
INCORRECT - TRY AGAIN ... !D
INCORRECT - TRY AGAIN ... !A
CORRECT
```

A small extract from sample run of the multiple choice test, 'GDMULT'.

Two multiple-choice test programs

A second type of test program – GDMULT – was developed from an existing program written by Ray Garnett. It is a traditional multiple-choice test in fact. Students go on-line as above and run the program GDMULT. They are offered a choice of tests. Having selected their test, students are then asked a series of questions to which there are 4 possible answers – A, B, C or D. A test may consist of up to 30 questions altogether (usually sufficient to cope with). Setting up this test is very easy. Data files are created in the EDITOR, as for GDTEST. Each question may consist of a maximum of 115 characters, including blanks. Each of the 4 alternative answers is also subject to this limitation. Thus the layout of a typical question plus alternative answers looks like this:

Question
Correct answer
Alternative wrong answer
Alternative wrong answer
Alternative wrong answer

Note that the correct answer always appears immediately after the question. This does not, however, mean that answer A is always the correct one, as the program jumbles the answers randomly.

When the program runs, the questions are presented in turn, the student being invited to select from alternatives A-D. If he gives a wrong answer he may try a second, third and fourth time, if necessary. At the end of the test these incorrectly answered questions are presented again.

This is where the clever bit occurs: the program then jumbles the alternative answers once more, thus making it pointless for a student to remember, say, that the correct answer to question 7 was alternative A. He must attempt to remember the content of the answer rather than its alphabetical designator. A student cannot escape from a test until he has given first-time correct answers to *all* questions, either on the first or subsequent runs. At the end of the test a score is given, expressed as the percentage of first-time correct answers out of all questions presented, and the time taken to complete the test is indicated. The computer stores the student's result and then offers a new test. If a new test is accepted, then the process described above starts again. GDMULT is excellent for short questions and very useful as a speed exercise.

A third testing program has been developed by John Muir. This is again a traditional multiple-choiced test, but designed for use outside computer workrooms, away from interactive terminals.

Questions and alternative answers are

set out on sheets of paper and handed out to students in class. Each student also receives a computer-readable card. On this card he marks in black pencil little slots corresponding to the questions and the alternative answers he considers correct. He adds his initials, for identification purposes, at the far right of the card and simply hands in the completed card to the teacher. The cards are then fed into the computer, together with a master card prepared previously, against which all the student cards are checked. The master card is accompanied by a parameter card on which information relating to the number of questions (maximum 35) and the number of alternative answers is indicated. The resulting output from the computer is in the form of a list of all students taking the test, with an indication of the questions they got wrong, their mark expressed as a percentage, and a group profile, showing which questions the group as a whole answered correctly or incorrectly.

This is a very flexible test. Up to 9 alternative answers to each question may be given, providing this figure is constant within each test. It is a particularly useful test if large numbers of students are involved. Additionally, there is no need for all students taking the test to be present in the same room at the same time.

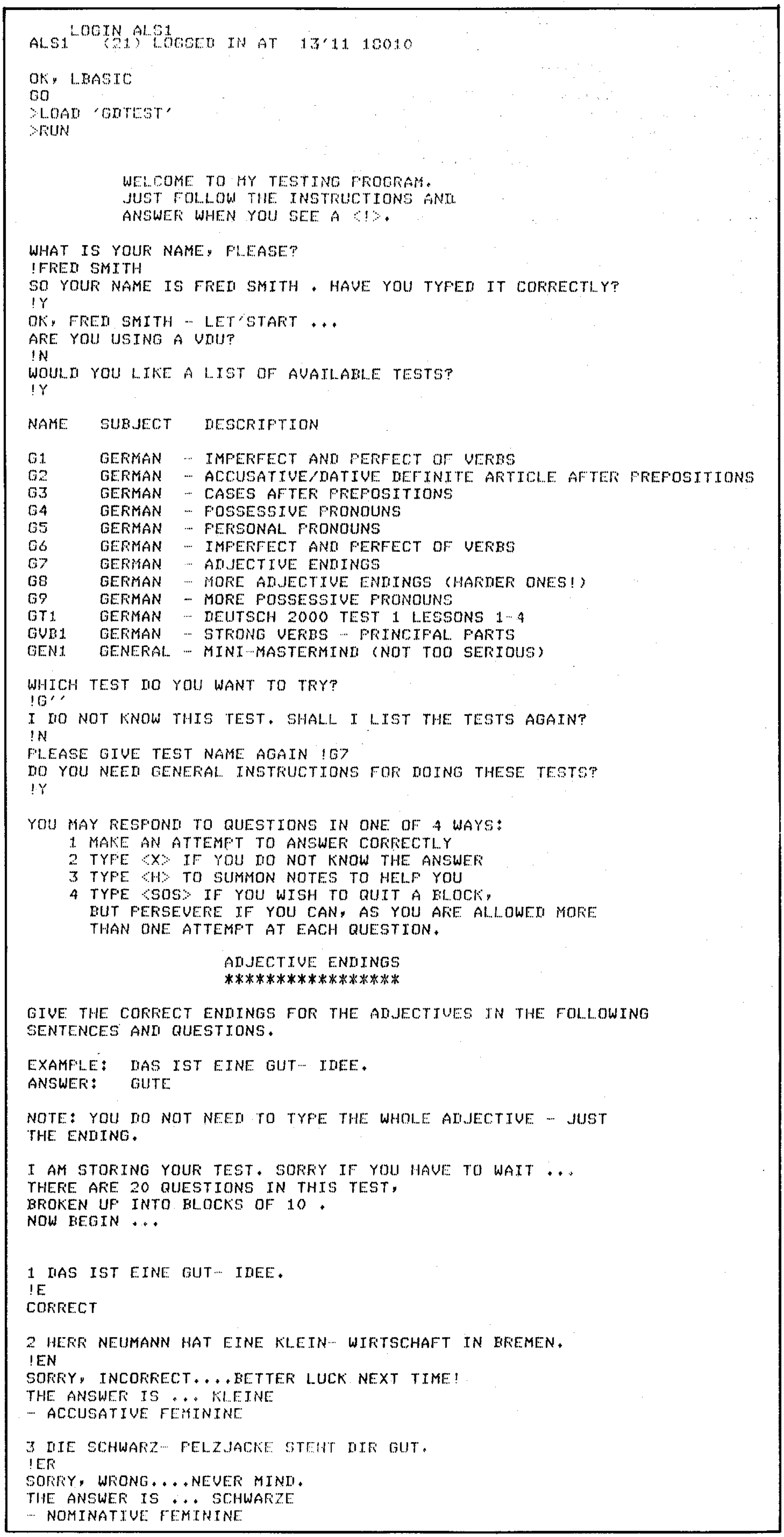
Conclusions

This, then, is where we have arrived so far in the sphere of CAL at Ealing College of Higher Education in the teaching of modern languages. Students who have acted as guinea pigs in trying out the programs have generally been very positive in their reaction to the application of new technology to language learning. So far we have developed only a trial set of programs and data files. Ideally, we hope to write a complete package of language tests and drills to be used at all levels. What we have used on the Prime 300 is equally applicable to microcomputers, of which increasing use is being made (lamentably, mainly in the scientific sphere) in schools and FE colleges. If our package can be adapted for use on a microcomputer it would soon reach a wider audience. It would be nice to be able to say that Ealing will be at the forefront of language teaching by microcomputer, in the same way that we spearheaded the movement towards language laboratories in the 1960s, but it is mainly a lack of financial support which is holding things up at present.

The scope for new developments is enormous. So far we have only begun to scrape the surface, and it is painfully clear that we lag behind the USA in CAL by almost a decade. Linking a tape-recorder to a microcomputer is a future possibility, facilitating dictation tests or aural comprehension tests. More sophisticated remedial and guidance procedures within programs would ensure that no student attempts anything way beyond his ability and that all errors are backed up by branching to appropriate advice and further testing.

At least one company has begun work on the computerised language laboratory. This is not merely a dream of the future, as work on voice input and output is already at an advanced stage. It will almost certainly not be long before the voice in the language laboratory headphones stops its inexorable, unsympathetic rambling and says, 'I'm sorry, I didn't quite catch that. Let's try that last block of questions again'. ■

Graham Davies is a lecturer at the Ealing College of Higher Education.



```
         LOGIN ALS1
ALS1    (21) LOGGED IN AT  13'11 10010

OK, LBASIC
GO
>LOAD 'GDTEST'
>RUN


         WELCOME TO MY TESTING PROGRAM.
         JUST FOLLOW THE INSTRUCTIONS AND
         ANSWER WHEN YOU SEE A <!>.

WHAT IS YOUR NAME, PLEASE?
!FRED SMITH
SO YOUR NAME IS FRED SMITH . HAVE YOU TYPED IT CORRECTLY?
!Y
OK, FRED SMITH - LET'START ...
ARE YOU USING A VDU?
!N
WOULD YOU LIKE A LIST OF AVAILABLE TESTS?
!Y

NAME   SUBJECT   DESCRIPTION

G1     GERMAN  - IMPERFECT AND PERFECT OF VERBS
G2     GERMAN  - ACCUSATIVE/DATIVE DEFINITE ARTICLE AFTER PREPOSITIONS
G3     GERMAN  - CASES AFTER PREPOSITIONS
G4     GERMAN  - POSSESSIVE PRONOUNS
G5     GERMAN  - PERSONAL PRONOUNS
G6     GERMAN  - IMPERFECT AND PERFECT OF VERBS
G7     GERMAN  - ADJECTIVE ENDINGS
G8     GERMAN  - MORE ADJECTIVE ENDINGS (HARDER ONES!)
G9     GERMAN  - MORE POSSESSIVE PRONOUNS
GT1    GERMAN  - DEUTSCH 2000 TEST 1 LESSONS 1-4
GVB1   GERMAN  - STRONG VERBS - PRINCIPAL PARTS
GEN1   GENERAL - MINI-MASTERMIND (NOT TOO SERIOUS)

WHICH TEST DO YOU WANT TO TRY?
!G''
I DO NOT KNOW THIS TEST. SHALL I LIST THE TESTS AGAIN?
!N
PLEASE GIVE TEST NAME AGAIN !G7
DO YOU NEED GENERAL INSTRUCTIONS FOR DOING THESE TESTS?
!Y

YOU MAY RESPOND TO QUESTIONS IN ONE OF 4 WAYS:
    1 MAKE AN ATTEMPT TO ANSWER CORRECTLY
    2 TYPE <X> IF YOU DO NOT KNOW THE ANSWER
    3 TYPE <H> TO SUMMON NOTES TO HELP YOU
    4 TYPE <SOS> IF YOU WISH TO QUIT A BLOCK,
      BUT PERSEVERE IF YOU CAN, AS YOU ARE ALLOWED MORE
      THAN ONE ATTEMPT AT EACH QUESTION.

                   ADJECTIVE ENDINGS
                   *****************

GIVE THE CORRECT ENDINGS FOR THE ADJECTIVES IN THE FOLLOWING
SENTENCES AND QUESTIONS.

EXAMPLE:  DAS IST EINE GUT- IDEE.
ANSWER:   GUTE

NOTE: YOU DO NOT NEED TO TYPE THE WHOLE ADJECTIVE - JUST
THE ENDING.

I AM STORING YOUR TEST. SORRY IF YOU HAVE TO WAIT ...
THERE ARE 20 QUESTIONS IN THIS TEST,
BROKEN UP INTO BLOCKS OF 10 .
NOW BEGIN ...


1 DAS IST EINE GUT- IDEE.
!E
CORRECT

2 HERR NEUMANN HAT EINE KLEIN- WIRTSCHAFT IN BREMEN.
!EN
SORRY, INCORRECT....BETTER LUCK NEXT TIME!
THE ANSWER IS ... KLEINE
- ACCUSATIVE FEMININE

3 DIE SCHWARZ- PELZJACKE STEHT DIR GUT.
!ER
SORRY, WRONG....NEVER MIND.
THE ANSWER IS ... SCHWARZE
- NOMINATIVE FEMININE
```

A small extract from 'GDTEST', the main testing program.