VOL.2
NO. 4

# STATUS 1500

MAY
1984

* 12 issues £10.50 (UK) * overseas £14.50 * published monthly *

* Edited by RONALD COHEN, 62 BLENHEIM CRESCENT, LONDON W.11 *

## MORE APOPLOGIES

The strain of getting 2 issues out within a fortnight has inevitably led to some imperfections. There may be more than the usual number of typographic errors, and uneveness of layout. Most of all I must again apologise for leaving out contributions from subscribers which I had looked forward to printing. These apologies are directed both to those contributors who may feel their efforts are not appreciated (indeed they are appreciated) and to the many subscribers who complain that there is too much talk, and not enough programs and technical information.

Time is the problem. A page of waffle takes not long to prepare. I can write it in the train, and the only editing may be the correction of a few mispellings, or a slight adjustment to the length; and even this can often be dealt with by closer or wider setting of the lines. Readers programs are less flexible. In many cases even identification is a problem. A number of strips of paper must be pieced together: often time-wasting researches must be made to rediscover the authorship. (Why oh why are you so shy about putting your names on your programs?) The program must be laboriously keyed in, keying mistakes debugged: the program itself must sometimes be debugged: it must be rigorously tested. Then it must be LLISTed: to key it in just as text is not quite safe: typing errors could creep in, whereas if the program is LLISTed immediately after it has been RUN, then one source at least of error is obviated. If it is printed on the Juki, space can be wasted. If printed by the CE 150, careful measurements must be made. Printed: cut up: stuck down: copied: marks Typexed out: recopied. No short cuts - and that is when all goes well, without snags. When things start to go wrong, as so often they can do, I sometimes feel that it is a miracle that the newsletter ever appears at all.

## CONTENTS

## SIGNALS

ALLAN THOMAS sends from New Zealand the latest issue (no.5) of PC 1500 USER. It contains 'Pontoon', an Adventure game, and some utilities in machine-code. The group now has 13 members.

A splendid publication. Best of luck.

JOHN WARNER notes that his prize will be "carefully selected for cheapness, uselessness, and grotesque design". (see MINDBOGGLE last month). He says that he confidently expects a signed photograph of the editor.

No comment. The editor is - for once - speechless.

ARTHUR COX points out that if a line ends with a quotation mark, this may be omitted without harm, thus saving space.

This could occasionally be useful. But it leads to messy, inconsistent programming. How for instance would you distinguish between GOTO " " and GOTO"" ?

ARTHUR COX persists that the technique can be useful. He adds irrelevantly that he never has a line with nothing in it.

The editor, who is equally obstinate, maintains that this sort of sloppy programming makes editing difficult. It adds, for instance, to the problems involved in Variable Changing (see PEEK & POKE). He notices that Mr COX does not omit the fullstop at the end of a sentence, even when this ends a paragraph. We have been strictly taught to read and write, but as far as programming is concerned, we are self-educated. We have the awkward task of trying to maintain and create standards, not destroy them.

M.J.SMITH expresses his sympathy with the plight described on page 21, but believes that I may be mistaken about the cause of the trouble. He points out that the PC 1500 will run without any batteries at all while it is attached to the CE 150.

This invalidates my suggestion that the batteries might have enough power to drive other peripherals, but not the CE 158 /printer. But the difficulty occurred while the PC 1500 was attached directly to the CE 158; and exhaustive tests, many more than were briefly described, proved conclusively that low batteries in the computer caused the trouble. There could of course have been contributory factors, such as the charge of the CE 158 for instance. But with the set-up I was working with, I could not get the printer to print with the old batteries in the computer; and with new batteries everything worked OK. Your suggestion that one should occasionally 'roll' the batteries in the computer, to break up the formation of any film of contamination, is a useful hint.

R.WILLIAMES asks whether the substitution of a 4Mhz crystal in the computer upsets the CE 158?

Probably. Its all a magical electronic mystery to me. Has any reader dared to experiment along these lines?

CHIA PIA LON writes from Malaysia to ask about POKEing machine-code routines into memory locations. What happens if one POKEs into ROM, or into an address with which one should not interfere?

You cannot POKE into ROM. This stands for 'READ-ONLY MEMORY' and means what it says. If you POKE into a wrong address in RAM you will do no permanent damage, but obviously you can destroy your program or data if you POKE into pointers or buffers. At the worst, you may create an endless loop, even temporarily disabling the ON/BREAK key. You may have to RESET: you might only be able to turn the computer off by taking the batteries out; and owing to the non-volatile memory, it can take quite a few minutes before they should be put back.

If you have been diligently reading this series, you have been wasting your time. It is necessary to work through it: draw up your own scheme of attack, if you don't like mine: try writing a few lines of program. Perhaps this may come more easily when the series is further advanced.

But I promised that this month we would start coding. So here we go with section 3, as suggested. Lets actually START!

```
    3000: INPUT "HOW MANY PLAYERS";H
    3010: FOR F=1 TO H
    3020: INPUT "name?";A$(F): INPUT"capital?";A(F): B=B+A(F) §B is bank's capital
    3030: NEXT F
    3040: INPUT "alter banks capital to: ";B §if not to equal total
                                               of players' capital.
```

So lets just RUN this section. And almost immediately:

```
    ERROR 6 IN 3020
```

We have gone wrong right from the start. We cannot use DIMensioned variables until we have DIMensioned them. So we will do so - provisionally. Changes can me made later, if required.

```
    1000: DIM A$(7), A(3,7) §more than suggested originally
```

We are taking 7 as the maximum number of players, for the moment. The game would be too slow with more. An alternative would be to DIMension only for the actual number of players: but then suppose someone wants to join in later? Lets try this first.

Now to choose the winning number, before bets are actually placed.

```
    5000: PAUSE "PLACE YOUR BETS!"
```

§This is not section (d),'Spinning the Wheel'. Remember we said we would secretly choose the winning number before betting starts.

```
    5010: FOR F=1 TO RND 37:READ W: NEXT F
   60000: DATA 0,1,2,3,.....17,18
   60010: DATA 19,20,21,....35,36
```

Why not just pick a random number: W=RND 37-1? Two reasons: firstly, we may wish to change the DATA to the order numbers appear on the wheel: secondly, we might want 2 zeros, as in some casinos in USA. Note that we are putting the DATA at the end. The idea of DATA at the start of the program dates from the early days of computers, before INPUT was invented, and values could only be inserted by altering DATA in the program. Also in most computers the start of the program is easiest to access. With the PC 1500, it is equally easy to access either end of the program.

At this point, looking back, change line 3000: INPUT "HOW MANY PLAYERS?";H to 3000: INPUT "how many players?";H. If we keep all INPUTs in lowercase, and all other displays in capitals, it is more consistent: and this can be mildly helpful. Also, leave a space after the INPUT prompt. It makes whatever is INPUT separate, and easier to read.

We have not got very far. But we must tidy up a little before going further. Remember we said "Clear alphabetic labels"? So lets put them in. Lowercase will stand out best here.

```
    1000: "initialise" DIM......
    3000: "players" INPUT......
    5000: "stakes" PAUSE ......
   60000: "wheel" DATA .......
```

One last task this month: make a chart of the variables you have used, and what you have used them for, and whether global or local - i.e. whether you can use the same variable again for another purpose, or whether you would destroy essential data by doing so. Next month ......

# CRICKET SCORING

by J.R.Herring

*[This program provides a much more detailed analysis of a match than does an ordinary scorecard. A number of questions are asked at the beginning, and after each ball is bowled It is essential to answer each question. The CE 150 is required. The detailed account which is printed out makes following a match, whether at the ground, or over the radio, even more enjoyable.]*

STATUS 1= 5287

```
200 "C":CLEAR :LF 2:LOCK :CSIZE 2
202 LPRINT "Cricket Scoring"
204 REM J.R.Herring..September 1982
206 DIM B$(12)*10,F$(12)*10,F(12),W(12),L$(12)*30,L(12),M(12),D(12)
208 "AA":INPUT "TYPE SIZE.1 OR 2...";C
210 IF (C=1)+(C=2)<>1GOTO "AA"
212 CSIZE C
214 "BB":INPUT "COLOUR..0,1,2,3 .....";C
216 IF (C=1)+(C=2)+(C=3)+(C=0)<>1GOTO "BB"
218 COLOR C
220 INPUT "HOME TEAM....";H$
222 INPUT "VISITORS....";V$
224 LPRINT H$;" v ";V$
226 "CC":INPUT "WHICH TEAM TO BAT- H/V?";W$
228 IF (W$="H")+(W$="V")<>1GOTO "CC"
230 LF 1
232 IF W$="H"LET B$(0)=H$
234 IF W$="V"LET B$(0)=V$
236 N=1:WK=1:P=1
238 LPRINT B$(0);" WILL BAT"
240 A=1000:DZ=200
242 LF 1:LPRINT "How many runs do"
244 LPRINT "they need? If not":LPRINT "batting last ENTER"
246 INPUT "RUNS NEEDED TO WIN  ";A
248 LF 1:IF A<1000LPRINT "RUNS NEEDED";A
249 USING "###.##":INPUT "Limited Over? How many?";DZ:IF DZ<200LPRINT A/DZ;" runs per over.
250 LF 1:USING
252 INPUT "BATSMAN NO. 1      ";B$(1):GOSUB "DD":GOSUB "EE":GOTO "FF"
254 "FF":S$=B$(1):N$=B$(2)
256 INPUT "LIST FIELDING TEAM: No.1 ";F$(1)
258 INPUT "FIELDING TEAM,No.2    ";F$(2)
260 INPUT "FIELDING TEAM,No.3    ";F$(3)
262 INPUT "FIELDING TEAM,No.4    ";F$(4)
264 INPUT "FIELDING TEAM,No.5    ";F$(5)
266 INPUT "FIELDING TEAM,No.6    ";F$(6)
268 INPUT "FIELDING TEAM,No.7    ";F$(7)
270 INPUT "FIELDING TEAM,No.8    ";F$(8)
272 INPUT "FIELDING TEAM,No.9    ";F$(9)
274 INPUT "FIELDING TEAM,No.10    ";F$(10)
276 INPUT "FIELDING TEAM,No.11    ";F$(11)
278 INPUT "TWELFTH MAN     ";F$(12)
280 LF 1
282 "HH":INPUT "START BOWLING Y/N  ";W$
284 IF W$="Y"GOTO "GG"
286 IF W$<>"Y"GOTO "HH"
288 LF 1
290 "GG":INPUT "BOWLER IS NUMBER...";M
292 IF M<1GOTO "GG"
294 IF M>12GOTO "GG"
296 LF 1:LPRINT "BOWLER IS NUMBER";M;" ";F$(M):V=V+1:F=F(M)
298 LPRINT "OVER NO.";V
300 "JJ":R=0
302 ON ERROR GOTO "II"
304 "II":X=0:INPUT "1st.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JR"
306 GOTO "II"
308 "JR":G=0:K=.1:ET=0:GOSUB "DJ":IF ET=1GOTO "II"
310 GOSUB A$:ON ERROR GOTO "II"
312 IF G=1GOTO "JJ"
314 R=0:ON ERROR GOTO "KK"
316 IF T>=AGOTO "LL"
318 "KK":X=0:INPUT "2nd.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JS"
320 GOTO "KK"
322 "JS":G=0:K=.2:ET=0:GOSUB "DJ":IF ET=1GOTO "KK"
324 GOSUB A$:ON ERROR GOTO "KK"
326 IF G=1GOTO "KK"
328 R=0:ON ERROR GOTO "MM"
330 IF T>=AGOTO "LL"
332 "MM":X=0:INPUT "3rd.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JT"
334 GOTO "MM"
336 "JT":G=0:K=.3:ET=0:GOSUB "DJ":IF ET=1GOTO "MM"
338 GOSUB A$:ON ERROR GOTO "MM"
340 IF G=1GOTO "MM"
342 R=0:ON ERROR GOTO "NN"
344 IF T>=AGOTO "LL"
```

```
346 "NN":X=0:INPUT "4th.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JU"
348 GOTO "NN"
350 "JU":G=0:K=.4:ET=0:GOSUB "DJ":IF ET=1GOTO "NN"
352 GOSUB A$:ON ERROR GOTO "NN"
354 IF G=1GOTO "NN"
356 R=0:ON ERROR GOTO "OO"
358 IF T>=AGOTO "LL"
360 "OO":X=0:INPUT "5th.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JV"
362 GOTO "OO"
364 "JV":G=0:K=.5:ET=0:GOSUB "DJ":IF ET=1GOTO "OO"
366 GOSUB A$:ON ERROR GOTO "OO"
368 IF G=1GOTO "OO"
370 R=0:ON ERROR GOTO "PP"
372 IF T>=AGOTO 571
374 "PP":X=0:INPUT "6th.  Runs,NB,WD,LB,B,WK ";A$:GOTO "JW"
376 GOTO "PP"
378 "JW":G=0:K=1:ET=0:GOSUB "DJ":IF ET=1GOTO "PP"
380 GOSUB A$:ON ERROR GOTO "PP"
382 IF G=1GOTO "PP"
384 LF 1
386 IF T>=AGOTO "LL"
388 LPRINT "End of over no.";V:X=T-U:ON ERROR GOTO 0
389 LPRINT TIME
390 LPRINT "Total from over  ";X:U=T
391 USING "###.##":LPRINT "Run rate =";T/V;" an over":IF A<1000LPRINT "Rate req.";(A-T)/(DZ-V)
392 LF 1:USING
394 M(M)=M(M)+(F=F(M)):D(M)=D(M)+K:GOSUB "QQ"
396 GOTO "RR"
398 "QQ"LPRINT F$(M);D(M);" overs:";M(M);" maidens:";F(M);" runs:";W(M);" wickets."
400 RETURN
402 "RR":LPRINT B$(0);T;" for";WT;" wickets"
404 LPRINT "EXTRAS";E
406 LPRINT S$;" not out";S
408 LPRINT N$;" not out";Y:IF T>=AGOTO "LL"
410 GOSUB "EX":IF V=DZ:GOTO "ZX"
411 GOTO "GG"
412 "DF":LPRINT "*** ALL OUT ***"
414 LPRINT B$(0);" all out for ";T:GOTO "SS"
415 "ZX"LPRINT B$(0);" overs completed for";T:GOTO "SS"
416 "D":LPRINT B$(0);"  declared at";T;" for";WT:GOTO "TT"
418 "LL":LPRINT B$(0);" won";T
420 "TT":M(M)=M(M)+(F=F(M)):D(M)=D(M)+K:GOSUB "QQ"
422 INPUT "Did last ball dismiss?";W$
424 IF W$="Y"GOTO "UU"
426 L$(P)=N$+" not out":L(P)=Y:P=P+1:GOTO "DH"
428 "UU":GOSUB "EX"
430 "DH":L$(P)=S$+" not out":L(P)=S:P=P+1
432 GOTO "VV"
434 "DD":LPRINT "BAT No.";N;"..";B$(N):RETURN
436 "SS":Z$=" not out"
438 L$(P)=N$+Z$:L(P)=Y:P=P+1
440 GOTO "VV"
442 "EE":N=N+1
444 INPUT "NEXT BATSMAN   ";B$(N):GOSUB "DD":RETURN
446 "1":R=1:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
448 "2":R=2:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
450 "3":R=3:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
452 "4":R=4:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
454 "5":R=5:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
456 "6":R=6:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN
458 "R":INPUT "Number of runs?";R:IF X=1LET R=0:S=S+R:F(M)=F(M)+R:T=T+R:GOSUB "S":GOSUB "X":RETURN

460 "0":R=0
462 GOSUB "S":RETURN
464 "EX":D$=S$:S$=N$:N$=D$:D=S:S=Y:Y=D:RETURN
466 "S":LPRINT S$
468 IF (A$="R")+(A$="NB")+(A$="WD")+(A$="B")+(A$="LB")+(A$="WK")<>1GOTO "WW"
470 LPRINT A$;R:GOTO "XX"
472 "WW":LPRINT " ";R
474 "XX":RETURN
476 "X":INPUT "Did batsmen cross? Y/N ";W$
478 IF W$="Y"GOSUB "EX"
480 RETURN
482 "NB":INPUT "How many runs from bat?";R:IF R>0GOTO "ZZ"
484 T=T+1:E=E+1:NB=NB+1:R=1:GOTO "DE"
486 "ZZ":S=S+R:F(M)=F(M)+R:T=T+R:X=1
488 "DE":G=1:GOSUB "S":GOSUB "X"
490 RETURN
492 "WD":INPUT "How many <WIDES>?";R:T=T+R:WD=WD+R:E=E+R:G=1:GOSUB "S":GOSUB "X":RETURN
494 "LB":INPUT "How many LEG-BYES?";R:T=T+R:LB=LB+R:E=E+R:GOSUB "S":GOSUB "X":RETURN
496 "B":INPUT 'How many BYES?";R:T=T+R:E=E+R:BY=BY+R:GOSUB "S":GOSUB "X":RETURN
```

```
498 "WK":INPUT "Was it from a NO BALL Y/N";W$:WT=WT+1
500 IF W$="Y"GOSUB "NB"
502 INPUT "Was it a RUN-OUT?.Y/N";W$:IF W$="N"LET W(M)=W(M)+1
504 IF R>0GOSUB "R"
506 INPUT "How  OUT ???";Z$
508 INPUT "Was it strikers wicket?";W$
510 IF W$="N"GOSUB "EX"
512 L(P)=S
514 L$(P)=S$+" "+Z$
516 LPRINT L$(P);L(P):P=P+1
518 IF WT=10GOTO "DF"
520 GOSUB "EE":S$=B$(N):S=0
522 INPUT "Is new bat taking srike?";W$
524 IF W$="N"GOSUB "EX"
526 RETURN
528 "VV":LPRINT B$(0);" Innings":Q=P
530 FOR P=0TO Q
532 LPRINT L$(P);L(P)
534 NEXT P
536 LF 1:LPRINT "NB";NB
538 LPRINT "WD";WD
540 LPRINT "LB";LB
542 LPRINT "B";B
544 LPRINT "EXTRAS";E
546 FOR M=1TO 11
548 IF D(M)=0GOTO "DG"
550 GOSUB "QQ"
552 "DG"NEXT M
554 END
556 "DJ":ET=0:IF (A$="1")+(A$="2")+(A$="3")<>1GOTO "DK"
558 RETURN
560 "DK":IF (A$="4")+(A$="5")+(A$="6")<>1GOTO "DL"
562 RETURN
564 "DL":IF (A$="0")+(A$="B")+(A$="LB")<>1GOTO "DM"
566 RETURN
568 "DM":IF (A$="NB")+(A$="WK")+(A$="WD")<>1GOTO "DN"
570 RETURN
572 "DN":ET=1:RETURN
```

## PEEK POKE & MEMORY - XIV
### - the variable theme continued -

*The routine below is a slight elaboration of the ideas discussed last month (page 23). It was used to transform 'CRICKET SCORING' in this issue. Variable I was changed to C, and O to D, since the typeface used does not distinguish between letter O and number 0.*

*600: X is flag for inverted commas. Q is flag for REM (which might include quotes) and overides X.*
*615: G reproduces line number being dealt with.*
*620: flags Q and X reset at start of each line.*
*630: flag Q is set by REM.*
*640: flag X set and reset by inverted commas. PEEK (F-1) checks for Reserved Words, which of course must not be altered.*
*650 and 660: POKE C for I and D for O, and LPRINT line number where change occurs.*

```
600 X=1:Q=1:WAIT 0
610 FOR F=STATUS 2-STATUS 1+3TO STATUS 2-291
615 IF PEEK F=13LET G=256*PEEK (F+1)+PEEK (F+2)
620 IF PEEK F=13LET X=1:Q=1:F=F+3:PRINT G:GOTO 700
630 IF PEEK F=171AND PEEK (F-1)=241LET Q=0:GOTO 700
640 IF PEEK F=34LET X=-X:GOTO 700
645 IF (PEEK F=73OR PEEK F=79)AND PEEK (F-1)>221GOTO 700
650 IF Q*X=1AND PEEK F=73POKE F,67:BEEP 1:LPRINT G
660 IF Q*X=1AND PEEK F=79POKE F,68:BEEP 1:LPRINT G
700 NEXT F:BEEP 7
```

## USER FRIENDLINESS

Users are awkward creatures . The help you offer them is either ignored, or resented. Take ERROR messages, for instance. The statement:

```
INVALID PARAMETER XJK6B
```

is hardly likely to be greeted with cries of joyous enlightenment. On the other hand, the friendlier:

```
SILLY ME!
PRESSED THE WRONG KEY!
DONT COMPLAIN
BUT TRY AGAIN!
```

may well be found tedious on this side of the Atlantic. Equally the simple and accurate expression:

```
IDIOT!
```

is likely to be found unhelpful. One solution emerges from the rigid use of Menus. A main menu offers a choice of secondary menus, for the several functions of the program: and each of these may have tertiary menus appended. A choice is indicated on the display: the user merely has to press the key beneath the function he selects. There is of course always a key which returns to the main menu. This is very much the professional approach, as exemplified in the very successful EASI- software. The program is a 'black box' and is not accessible to the user, who is not expected to possess any ability to program. This approach is recognised as 'user friendly'. I am perhaps eccentric in considering 'Big Brotherly' would be a better description.

In less skilful hands the system can be disastrous. The wretched user is lost in a labyrinth of menus: whatever he does, he arrives at a dead end. This may be because he is trying to use the functions of the program in a sequence which was not envisaged by the author. Of course, the user has not read the instructions, or if he has read them, he has not understood them. At the best, he has read the instructions, understood them, and forgotten them. Take an example. Supposing the user tries to print out the results of a function of the program before he has actually run that function. An arrogant programmer will say "No one in their senses would do that!" (ignoring the fact that by now even the most intelligent user is probably senseless, anyway). A friendlier programmer will be willing to admit that there could be a reason. The user might wish to inspect the format of the printout; he might wish to assure himself that no residue remains from the previous run. But the really professional programmer will construct his program in such a way that it will not fail, even if used in a sequence which he had not imagined. This takes hard and careful work, and patience, and skill.

My own preference is for programs which are command-driven. Instead of a tree of menus, the appropriate command will initiate any function from any place in the program. This of course requires the user to learn a number - perhaps a large number - of commands: the idea is not universally popular. But it does lead to the sort of program which is not impossible for the user to tinker with, and adapt to his own particular needs.

Because it all depends on what sort of user you are trying to be friendly to. Whether you are writing for an impatient business user, who wants results, quickly and without fuss - or for the amateur who may be more interested in the method than in the end-product: he may even not wish to use the whole program, but just to lift part of it to incorporate in a program of his own.

Nor must the most important consideration be forgotten: is your program actually of any use to anyone? I receive some very clever programs, which have every facility, and cover every eventuality: and the outcome is that their results could be got far more quickly and more easily by hand. It would be technically possible to program the PC 1500 and CE 150 to produce striped wallpaper. A superior programmer will allow the user to produce his own pattern - checks, or squiggles, or pretty pink roses. You and I are even more friendly to the common user, and will write something more worthwhile.

## MINDBOGGLE CORNER

Are you handy with a soldering iron? I'm not. SHARP's EA158C parallel lead which connects printer to interface is extremely short, and will not reach to the front of the printer. So less than confidently I decided to make up an extension lead. Although all were not necessary, I decided to use all 25 pins at each end: the lead would then be usable with any other arrangement. When the soldering was done, I connected up, and keyed out the usual deathless prose.

You may imagine my complacency when I heard the printer happily chattering away. You may also imagine my horror when I saw the result:

THG SWKCK CSOWO GOX KWOPGD OWGS THG LC[[ DOG

But a moment's reflection showed the source of the trouble, and it only took a few moments more to put it right. Can you do as well? Answers on postcards, please, by June 30th. Prize: state your preference from a) second-hand soldering iron, or b) second-hand German dictionary (see page 38 ) or c) neither.

The answer to the April problem doubtless eluded few of you. The missing word is of course 'TIME' and the occasion was the annual introduction of Summer Time.

Apologies to DAVID RIHOY, whose answer to the March Mindboggler I had misunderstood. He has filled in the details with the routine on the left. It is in 2 parts: one for saving, and one for reloading. My own not dissimilar solution to this genuine problem is on the right. The locations 16586, etc., are for unexpanded memory. 16586 is equivalent to STATUS 2-STATUS 1+5. The routine goes at the head of the program it is to control. When first RUNning the program, answer the DIM? and LEN*? prompts. After reloading, RUN, and just reply to the DIM? prompt by pressing the ENTER key. The previous DIMensions will be restored, together with the contents of the DIMensioned variables.

```
10: "S"CLEAR :P1=0
    :P2=0:S1=35:D1=5
20: DIM T$(D1)*S1:
    P1=PEEK 30873:
    P2=PEEK 30874
30: B=B1*256+P2: T$
    (1)="TESTING
    CSAVE/CLOAD IN
    A PROGRAM
40: CSAVE M"DATA";
    B,24576: END
50: "L"CLEAR:
    CLOAD M"DATA"
60: POKE 30873,95,
    226: A=P1: B=P2
70: POKE 30873,A,B
80: PRINT T$(1): END
```

```
 1: REM ABCD
 5: ON ERROR GOTO 30
10: X=-1: INPUT  DIM?";X:
    INPUT "LEN*?";Y
20: DIM A$(X)*Y: POKE
    16586, PEEK 30823,
    PEEK 30824, PEEK 30873,
    PEEK 30874
30: POKE 30873, PEEK 16588,
    PEEK 16589
```

manually:

```
CSAVE M 16581, 18431
```

reloading:

```
NEW
CLOAD M 16581
POKE 16581,0
POKE 30823, PEEK 16586, PEEK 16587
```

DAVID RIHOY points out that the ability to CSAVE M and to CLOAD M from within a program is an unexpected and very useful facility.

## NOTES ON THE 'TRAMSOFT TOOL-2'

Since WALTER SPIEDEL does not answer letters, it was with some anxiety that I sent him a draft for DM.371 early last month. I was agreeably surprised to receive exactly 15 days later, the TOOL-2 (Tape Operation). The packaging was rudimentary, and there was not even a cap for the pins. However I find that these can be protected by the cap from an ordinary 36-pin D-plug. The packet had been opened by Customs, but no duty or VAT was charged. Another subscriber has had the same experience.

The TOOL-2 is about the size of a packet of cigarettes, and plugs into the CE 150, or directly into the PC 1500. It has 2 sockets, EAR and MIC: so, if it is is used directly with the PC 1500, there is no Remote Control facility. For Remote Control it is necessary to use the TOOL-2 in conjunction with the CE 150.

An extensive manual in somewhat technical German is supplied. There is also a leaflet giving the main commands, and their functions, in English. In the manual are several flow-charts which indicate clearly the essential syntax for each command. These charts require no linguistic ability. All commands may be abbreviated. FLOAD, for instance may be written as FL. or as FLO. All commands are represented in memory by 2-byte tokens, around 225/130: and behave like other Reserved Words. Naturally they only take this status if the TOOL-2 is attached while they are written into a program. In general, they follow the pattern of their normal slow equivalents. The only exception I have found is with FSAVE V and FLOAD V. These correspond to PRINT # and INPUT # which can be used without file-names: but for the TOOL-2 versions, for data saving and loading, a file-name is essential.

The other exception also affects data saving and loading. Undimensioned 2-character variables are not accepted! This is something to do with the REDIMENSION facility of the TOOL-3. However the inconvenience is minimal, since it is no hardship to save the whole of memory, both program and data, by FSAVE M.

The speed of saving and loading makes this easy. About 3.3 seconds for 1K. Thus FSAVE is over 20 times as fast as CSAVE. The other happy facility is VERIFY: Unlike normal PRINT #, this will also VERIFY a saving of data.

I have had a much higher than usual proportion of ERROR 43 and ERROR 44. This has happened even when VERIFY has indicated correct saving. However the increase in speed allows one to duplicate any recording: and, even with this precaution, working is still 10 times faster than normal. This sensitivity to ERROR may be partly due to the combination with the OLYMPUS C100. But the system does mean that it is now fully portable: slightly bulging pockets now contain computer, TOOL-2, and cassette-recorder. One microcassette holds all my programs.

It is necessary to use some care and concentration when using the TOOL-2, since most operations disable the ON/BREAK key. So a false attempt to FLOAD or FSAVE or VERIFY may result in a sort of endless loop. Escape from this can perhaps be made by detaching the EAR connection. Otherwise RESET, or (not recommended) detaching the TOOL-2. You are warned that you should not attach or detach the TOOL-2 without first turning off the computer: but if the OFF key is disabled, what are you to do? The noises made by the FLOAD V process are alarming at first, but one soon gets used to them. It would seem that the dramatic increase in speed is achieved by working, not on the normal digital system, ('Zweiton-Prinzip'), but on an ANALOGUE system ('Phasencodierung'). This is slightly guess-work on my part. My dictionary, though weighty, does not cover all the technical terms I could wish. Possibly as a result of the situation described, and perhaps also due to the low batteries in the computer, as mentioned in last month's editorial, I have encountered occasional corruption of a few characters in memory; and it was necessary to trace their addresses, and POKE the correct characters into those locations.

Comments from the experience of other users will be most welcome.