
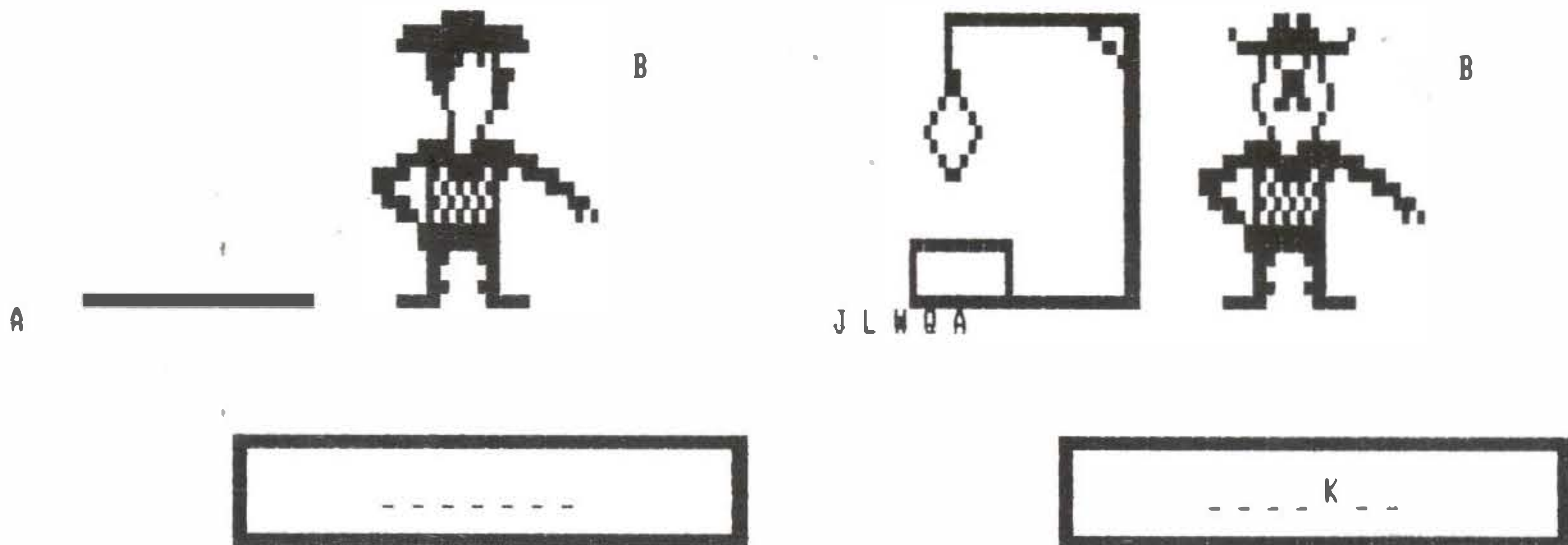


PRESS A LETTER KEY FOR YOUR GUESS
PRESS THE  KEY TO GUESS THE WORD



SUPER HANGMAN WITH GRAPHICS & SOUND

Also in this Issue:

HARDWARE:

Add Joysticks and Input/Output Points to your '80 — Part 5

PROGRAMMING:

The Theory and Techniques of Sorting — Part 3

SOFTWARE:

- Measurements
- Files
- Duplex — Double Precision Mathematical Functions
- Data Base Management System for Disk

MICRO-80

***** ABOUT MICRO-80 *****

EDITOR:	IAN VAGG
ASSOCIATE EDITORS:	
SOFTWARE LEVEL I :	MICHAEL SVENSDOTTER
SOFTWARE LEVEL II:	CHARLIE BARTLETT
HARDWARE :	EDWIN PAAY

MICRO-80 is an international magazine devoted entirely to the Tandy TRS-80 microcomputer and the Dick Smith System 80/Video Genie. It is available at the following prices:

	<u>12 MONTH SUB.</u>	<u>SINGLE COPY</u>
MAGAZINE ONLY	\$ 26-00	\$ 2-50
CASSETTE PLUS MAGAZINE	\$ 65-00	\$ 4-00 (cass. only)
DISK PLUS MAGAZINE	\$ 125-00	\$ 10-00 (disk only)

MICRO-80 is available in the United Kingdom from:

U.K. SUBSCRIPTION DEPT. 24 Woodhill Park, Pembury, Tunbridge Wells, KENT. TN2 4NW

Prices:	MAGAZINE ONLY	£ 16-00	£ 1-50
	CASSETTE PLUS MAGAZINE	£ 43-60	N/A
	DISK PLUS MAGAZINE	£ 75-00	N/A

MICRO-80 is available in New Zealand from:

MICRO PROCESSOR SERVICES, 940A Columbo Street, CHRISTCHURCH 1 N.Z. Ph. 62894

Prices:	MAGAZINE ONLY	NZ\$ 43-00	NZ\$ 4-00
	CASSETTE PLUS MAGAZINE	NZ\$ 89-00	NZ\$ 5-00
	DISK PLUS MAGAZINE	NZ\$ 175-00	NZ\$ 15-00

MICRO-80 is despatched from Australia by airmail to other countries at the following rates:

	(12 MONTH SUB.)	<u>MAGAZINE</u>	<u>CASS + MAG</u>	<u>DISK + MAG</u>
PAPUA NEW GUINEA	Aus\$ 40-00	Aus\$ 83-00	Aus\$ 143-00	
HONG KONG/SINGAPORE	Aus\$ 44-00	Aus\$ 88-00	Aus\$ 148-00	
INDIA/JAPAN	Aus\$ 49-00	Aus\$ 95-00	Aus\$ 155-00	
USA/MIDDLE EAST/CANADA	Aus\$ 55-00	Aus\$ 102-00	Aus\$ 162-00	

Special bulk purchase rates are also available to computer shops etc. Please use the form in this issue to order your copy or subscription.

The purpose of MICRO-80 is to publish software and other information to help you get the most from your TRS-80, System 80 or Video Genie and their peripherals. MICRO-80 is in no way connected with either the Tandy or Dick Smith organisations.

**** WE WILL PAY YOU TO PUBLISH YOUR PROGRAMS ****

Most of the information we publish is provided by our readers, to whom we pay royalties. An application form containing full details of how you can use your TRS-80 or System 80 to earn some extra income is included in every issue.

**** CONTENT ****

Each month we publish at least one applications program in Level I BASIC, one in Level II BASIC and one in DISK BASIC (or disk compatible Level II). We also publish Utility programs in Level II BASIC and Machine Language. At least every second issue has an article on hardware modifications or a constructional article for a useful peripheral. In addition, we run articles on programming techniques both in Assembly Language and BASIC and we print letters to the Editor and new product reviews.

**** COPYRIGHT ****

All the material published in this magazine is under copyright. That means that you must not copy it, except for your own use. This applies to photocopying the magazine itself or making copies of programs on tape or disk.

**** LIABILITY ****

The programs and other articles in MICRO-80 are published in good faith and we do our utmost to ensure that they function as described. However, no liability can be accepted for the failure of any program or other article to function satisfactorily or for any consequential damages arising from their use for any purpose whatsoever.

***** CONTENTS *****

	<u>PAGE</u>
EDITORIAL	2
PEEKING - FROM OUR U.K. CORRESPONDENT	3
JOYSTICKS AND INPUT/OUTPUT PORTS FOR YOUR '80 - Part 5	4
INPUT/OUTPUT - Letters To The Editor	5
THE THEORY AND TECHNIQUES OF SORTING - Part 3	6
<u>SOFTWARE SECTION</u>	
MEASUREMENTS.....L2/16K	12 & 22
SUPER HANGMAN.....L2/16K	13 & 24
FILES.....L2/48K	13 & 27
DATA BASE MANAGEMENT SYSTEM.....DISK/48K	15 & 29
DUPLEX.....L2/m.1.	16 & 31
MICRO-80 PRODUCTS	17
MICROBUGS	35
NEXT MONTH'S ISSUE	35
CASSETTE/DISK EDITION INDEX	36
ORDER FORM	36

MICRO-80 is registered by Australia Post — Publication SQB 2207 Category B
AUSTRALIAN OFFICE AND EDITOR:
 MICRO-80, P.O. BOX 213, GOODWOOD, SOUTH AUSTRALIA, 5034. TEL. (08) 211 7244
U.K. SUBSCRIPTION DEPT.
 24 WOODHILL PARK, PEMBURY, TUNBRIDGE WELLS, KENT TN2 4NW.

Printed by:

Shovel & Bull Printers, 312A Unley Road, HYDE PARK, S.A. 5061.
 Published in Australia by MICRO-80, 433 Morphett Street, ADELAIDE.

***** SPECIAL OFFER TO NEW READERS AND READERS RENEWING THEIR SUBSCRIPTION *****
***** SOFTWARE LIBRARY, VALUED AT OVER \$100 — FREE!!! *****

MICRO-80 has developed a new Library of Software consisting of 7 programs and a comprehensive user manual. The Software Library, on cassette, will be sent FREE to every new subscriber and to every subscriber who renews his subscription for another 12 months. Disk subscribers will receive their Software Library on a diskette. The new Software Library contains the following Level II/Disk Programs. All programs will also operate on the Model III.

Level I in Level II

Convert your Level II TRS-80 or System 80 to operate as a Level I machine. Opens a whole new library of software for your use.

Copier

Copies Level II System tapes, irrespective of where they load in memory. Copes with multiple ORG programs.

Z80 MON

A low memory, machine language monitor which enables you to set break points, edit memory, punch system tapes, etc...

Cube

An ingenious representation of the popular Rubick's cube game for Disk users.

Poker

Play poker against your computer, complete with realistic graphics.

Improved Household Accounts

Version 3.0 of this useful program. One or two bugs removed and easier data entry. This program is powerful enough to be used by a small business.

80 Composer

A music-generating program which enables you to play music via your cassette recorder and to save the music data to tape. This is an improved version of the program published in Issue 17 of Micro-80.

***** EDITORIAL *****

VIC-20 SALES SUSPENDED IN AUSTRALIA

Word has just been received that sales of the Commodore VIC-20 computer have been suspended in Australia because the mains transformer does not meet the power supply authorities' standards. Apparently the transformer has an unearthed metal case. It is not known how long this ban will last, presumably until Commodore can come up with a suitable replacement. In the meantime, the Dick Smith organisation is gearing up to change the transformer for an approved one, whilst the few enterprising dealers who grew tired of waiting for Commodore to produce the VIC-20 and imported them directly from the U.K. are still able to supply machines because the U.K. transformers meet the authorities' requirements.

BEARDING THE LION!!

Not long before Christmas, I was invited to attend a press conference in Sydney at which Tandy was to announce the Australian release of its TRS-80 Colour Computer. I was also promised that I could take home a computer to assist us in preparing a review. This was my first visit to Tandy headquarters in Sydney and was of considerable interest. No doubt many readers are wondering at my reception since there seems to be a general feeling that MICRO-80 is "anti" Tandy. Let me say at this point that MICRO-80 is not anti-Tandy. There are certainly areas in which we disagree with Tandy's policies and pricing but, overall, we want the TRS-80 to succeed and many of our critical comments are with marketing policies which sometimes seem to us to be holding back this excellent computer. Anyway, Tandy's executives and staff were extremely courteous and friendly and could not do enough to make us all welcome. Of the TRS-80 Colour Computer, you read last month. The other main feature of the day was a tour of the Tandy headquarters.

Tandy makes extensive use of Model II computers itself, particularly in word processing applications. We saw a new 100,000 word spelling checker in use with Scripsit, which will soon be available in Australia. We also saw the HOTLINE area where Tandy's staff answer technical queries from all over the country. The overall impression was of an organisation which is striving hard to support its products.

The visit to the warehouse and service section was enlightening. Being just before Christmas, the warehouse was far from full but the service section was chock-a-block with open Model III computers. Don't get the wrong impression - we are not suggesting that the Model III is unreliable. Apparently, Tandy Australia purchases Model III's from the U.S.A. as L1/4K versions and then upgrades them to L2/16K, adds disk drives etc. Which explains why the memory chips used in 48K Tandy machines are identical to those sold by MICRO-80 Products. Tandy staff explained that, as far as the Americans are concerned, Tandy Australia is just another customer and receives discounts on products in proportion to the quantity purchased. No doubt, this explains some of the difference in price between Australia and the U.S.A. By buying L1/4K Model III's and upgrade kits, Tandy Australia is able to minimise its costs.

The day finished with us being kitted out with a TRS-80 Colour Computer each (specially flown in from the U.S.A. for the occasion) and a large stock of ROMpak software for evaluation over the next month. The day was certainly well spent and enjoyable and I would like to take this opportunity to publicly thank Chuck Wise (Tandy's Managing Director) and his staff for their courtesy and helpfulness. I also extended an offer for a column in MICRO-80 to be written by Tandy. I believe that this offer has been accepted in principle and hope that, before long, we will start receiving regular contributions from Tandy telling us some of the inside news of TRS-80 development.

MORE SUPPORT FOR THE MODEL III (AND MODEL I)

Eddy Paay has been hard at work for several months now carrying out the necessary research work required to prepare a Model III version of his ROM REFERENCE MANUAL. The new version (which will probably have a different name, more descriptive of its true nature) caters for both Model I and Model III owners. It largely follows the format of the original version but includes much more information on the Model I and, of course, the same information on the Model III which is all new. The most exciting news, however, is that each manual will be supplied with Eddy Paay's own machine language Debug program, on cassette. Eddy could not find a commercially available debug program which he considered satisfactory, so he wrote his own. Features include the ability to single step through your program; a disassembler which disassembles the next instruction before executing it, or allows you to bypass execution and pass on through the program disassembling as you go; memory display and editing in Hex or ASCII; Register editing; the ability to punch out System tapes etc. etc. Eddy believes that as little of the screen as possible should be taken up by the Debug program so its display is confined to the bottom three lines of the screen, leaving the remainder for your own program. Debug runs in the TRS-80 Models I and III and the System 80/Video Genie, with or without disk drives.

The whole package manual and program will be available around the end of April for \$29.95 + \$1.15 p.p. Any orders received for the Level II ROM Reference Manual from now on will be held over until the new version is available.

REVISED FORMAT

This issue, you will notice that we have revised the format of MICRO-80 somewhat. We have slightly reduced the size of the print used for listings and have arranged them at the back of the magazine in two columns. The text for each program is now separated from its listing. You will notice that the index has two different page numbers for each program. The first number refers to the start of the text and the second to the start of the listing. The main reason for the changed layout is to allow us to fit more into each issue, particularly articles, reviews etc., in response to your requests. We hope you like this change and particularly the extra material it enables us to print each issue.

- 000000000 -

***** PEEKing (U.K.) by Tony Edwards *****

One of the more important aspects of the duties of a Correspondent is to engender contact and communication between readers in different parts of the world. In view of this, I have been very pleased to receive letters from so many MICRO-80 readers from both Europe and Australasia since I took up the post of U.K. Correspondent for MICRO-80. I have enjoyed reading your letters and I hope I have been able to help when help was requested. In this piece I repeat my offer to do anything I can to help readers anywhere in the world. If you think a contact in the U.K. could help you, just drop me a line at 23 Foxfield Close, Northwood, Middlesex, United Kingdom.

Another way that I can help the exchange of ideas is to reproduce some of the hints that are circulating in this country regarding the '80 micro-computer. In the early editions of this magazine I noticed that the tricks and hints reported from Australia were different to those circulating in this country, so I intend to devote my article now and again to the reporting of such things. I claim no originality for the hints reproduced, as they are all common knowledge here, but may be new to some readers.

Ever forgot the name of the program on an old tape? I have seen a number of very complex ways of loading an unnamed System tape, but the easiest way is to find its name by running the short program:- 10 INPUT#-1, A\$;?A\$ This should output the program name. If it was a System tape, the name will be preceded by U, and if it was a BASIC tape the name letter only is output. Data tapes produce the first item of data and Tiny Pascal tapes react as System tapes.

It has been mentioned to me that Disc users do not get many hints, so here is one for them. The Disk command CMD"E" is not in the Model I TRSDOS manual. If you type it after a Disk Error you will get useful further details of the error. In Disc Basic a NEWed program can be recovered by POKE 26811,1 then SYSTEM /11395. This will allow LISTING or CSAVEing but do not try RUNNING. Non-disk users should POKE 17130,1 for the same results.

There has been a heavy rumour that Tandy is to withdraw from the U.K. the sale of the Model I, due to its radio interference problems. This is not the case. If Tandy does withdraw, it will not be because of radio interference - the laws in the U.K. are not like the American ones. In America, where the Model I has been withdrawn, it is still legal to buy and sell secondhand Model I machines, and it seems that they are changing hands at prices above the original selling price. As it is still legal over there to sell add-ons for existing Model I's, I think that there will be support for that machine for some time. In any case, Lowe Electronics is prepared to support the TRS-80 by supplying Genie add-ons and a TRS-80/Genie converter.

Tandy has just started an in-house magazine for buyers of their products in an attempt to reverse the image of not caring for their users, which they have been getting in the U.K. The new free magazine is very good, if the first issue is anything to go on. In this issue, a new High Resolution Graphics mod. is introduced. The first of many, I guess. It uses no RAM and has a resolution of 348 x 192. Fitted by Tandy it costs £169.00.

LOWER CASE IN VIDEO GENIE/SYSTEM 80.

Attention all Video Genie owners, especially those who bought their machines after October 1981! If you do not have lower case fitted, have you ever RUN a lower case driver? If not, you may be in for a pleasant surprise. It seems that the latest Genies all have lower case, even though they were not sold as such. Just get a lower case driver (MICRO-80 Issue 18, page 29) and run it. If you are lucky, there is your lower case - Free. If you are unlucky, the driver will scramble your keyboard vectors so you will have to switch off to get them back.

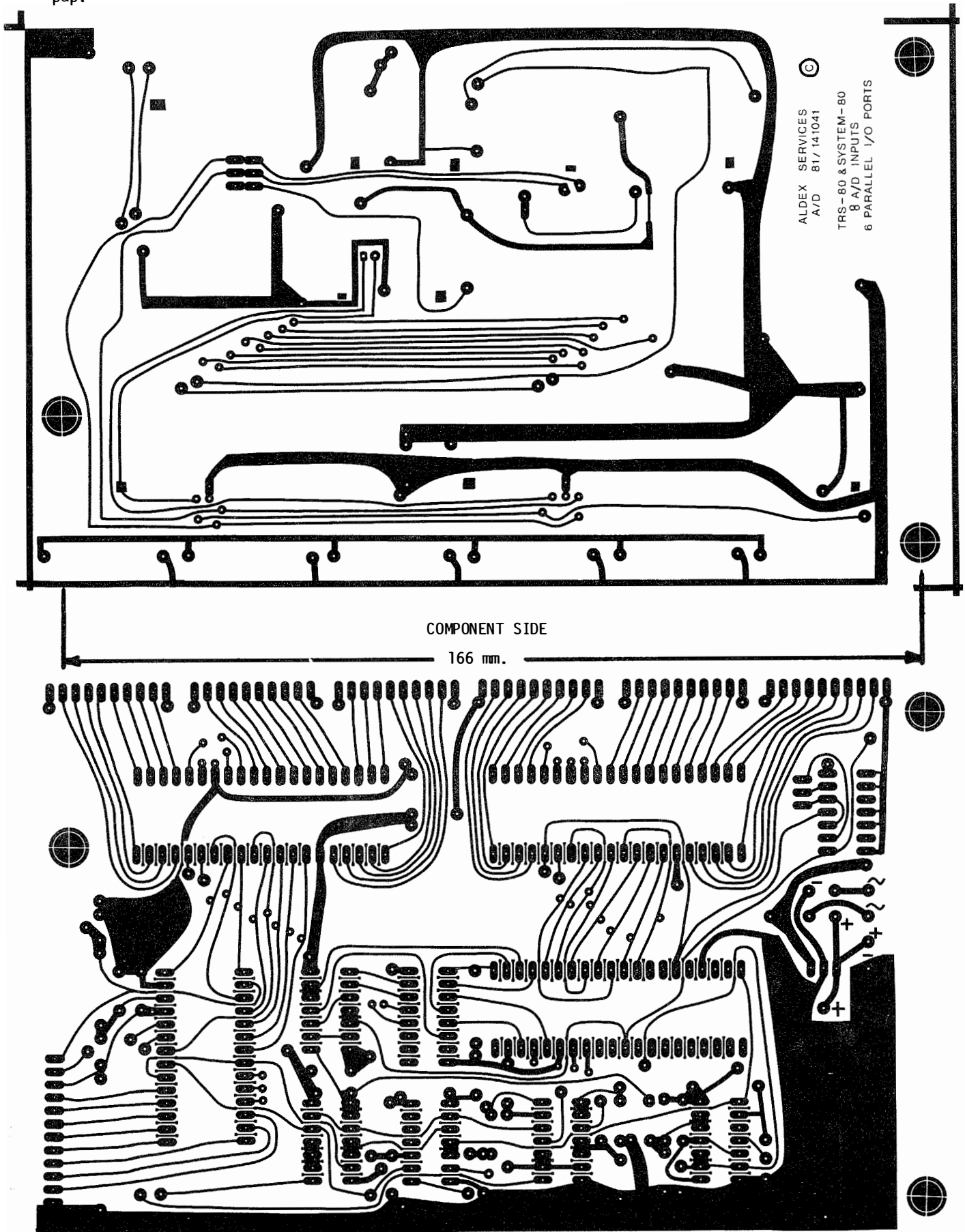
To finish, I have another rumour for spreading. Soon we will have another '80 to add to our camp. Not another version of the Genie, but a completely new computer. The new machine will be able to pretend that it is a TRS-80, or a PET or a number of other different types of computer. I do not know how it will be able to deal with the trick programs that are used, however, and perhaps it will only be able to use standard Microsoft Basic Software. (This is the Acorn Proton, designed to a specification prepared by the BBC - Ed.)

- 000000000 -

***** JOYSTICKS AND INPUT/OUTPUT PORTS FOR YOUR '80 by Alan Dent *****

PART 5 - P.C. BOARD TRACK LAYOUT

The PC board track layout is shown below at actual size. This should enable you to construct your own if you so wish. Otherwise, you may obtain a board from MICRO-80 for \$24 including p&p.



COMPONENT SIDE

166 mm.

TRACK SIDE

***** INPUT/OUTPUT *****

From: Martin Hodge, Victoria.

Many of your readers, I am sure, own lineprinters and have met the problems associated with trying to make neat columns on the right-hand side of the page. These problems are due to the '80's tab function being limited by the ROM software to 63 characters, the width of the VDU. The Level 2 reference manual states that if a TAB of greater than 63 is attempted then the TAB will occur on succeeding lines. However this is not quite correct. A TAB(65) will produce an identical effect as TAB(1), as will a TAB(129) and TAB(193).

Many 'Tricks' have been tried and everyone has their favourite. However this one is the simplest I have seen and I am sure it will interest many readers. It was discovered by Frank Ellett of MICRO HIRE, Palm Beach, Queensland and myself whilst exploring the Level 2 ROM.

The TAB routine uses a subroutine located at 2317H which basically takes the value from within the brackets of the TAB statement -TAB()- and stores this in the computer's ACC. area. Then using another subroutine at 2B05H this value is read into the DE registers. It is then ANDED with 3FH to eliminate the most significant bits and thus ensure that the resultant TAB is less than 64.

To now TAB beyond 63 it is only necessary to reread the actual value from the ACC. Conveniently soon after the ANDING the TAB routine CALLS a location in RESERVE RAM (41D3H). This location has a RETURN stored in it during system initialization. If instead we POKE a JUMP to the subroutine which read the value from the ACC then this will replace the ANDED value from the TAB statement with the value that is actually there. The required routine is located at 2B05H. Because the main routine CALLS 41D3H but we Jump to 2B05H the RETURN found at the end of 2B05H will return the computer directly to where we distracted it.

The required code can be POKED into the RAM during program initialization:-

```
10 POKE 16851,195 :POKE 16852,05 :POKE 16853,43
```

(JMP 2B05). The above line need only be entered at the start of the program requiring the extended TAB and will allow the printer to be TABBED up to 255 characters using normal syntax.

This modification does not seem to affect any other BASIC commands but careful testing should be done before including it in any "important" programs as no guarantee is given.

I look forward to receiving MICRO-80 each month and congratulate you on its presentation and content. I would however like to see better descriptions of what each PEEK and POKE is actually doing, and also associated circuit diagrams accompany any hardware modifications, not just pictorial drawings of "Where The Wires Go".

(Thank you for this contribution, Martin. It should be of considerable help to those readers using 80 character and wider printers.

We will endeavour to meet your request to explain more fully what each PEEK and POKE does in the programs we publish. To an extent, this depends on the authors supplying the programs, so authors, how about telling us in the documentation which accompanies each program? Your letter was written some time ago and I think you will agree that we have now mended our ways and publish circuit diagrams with hardware projects. -Ed.)

From: P.G. Smith, N.S.W.

I would like to congratulate you and your associates on a fine magazine and hope that your informative work will continue.

I am writing this letter to pass on to you and the magazine some helpful routines I have learned about, and which you may or may not know about.

BREAK KEY.

To disable the 'BREAK' key, execute the command: POKE 16396,241. Using the INKEY\$ function your program can check for the 'BREAK' being depressed. (It will return the value of 4).

To return the 'BREAK' key to normal operation execute the command: POKE 16396,201.

The TRS-80 can be reset to MEMORY SIZE when the 'BREAK' key is depressed, by using the command: POKE 16396,195.

NB. This is a useful hint if you wish to limit access to any particular program.

SLOW LIST FUNCTION.

If you find the scrolling speed during a LIST command is too fast, the following routine will slow it down:

```

10 FOR X = 16863 TO 16865
20 READ A: POKE X,A
30 NEXT X
40 FOR X = 32754 TO 32767
50 READ A: POKE X,A
60 NEXT X
70 DATA 195,242,127,58,128,56,31,208,197
80 DATA 1,0,32,205,96,0,193,201
90 END

```

You MUST protect 16K bytes (16K = 32751) and you MUST RUN this routine before loading your program. Once the routine has been executed, depress "SHIFT" to slow the LIST scrolling.

(Thank you for this contribution, Mr. Smith. Rod Stevenson has been exploring the disabling of the BREAK key in his series on Better BASIC Programming so readers now have a number of choices. The slow LIST function will be a great help to us all. - Ed.)

- 000000000 -

***** THE THEORY AND TECHNIQUES OF SORTING - PART 3 by B. Simson *****

This month I shall be demonstrating the techniques of sorting "by insertion", in particular, the ranking sort, also known as the insertion sort.

GENERAL DESCRIPTION.

In general, the insertion sort involves comparing or "ranking" an item in an array with items further up the array, which have already been sorted into the correct sequence, but may not yet be in their final positions. When the right insertion position is found, the item being processed is inserted at that position and processing continues with the next item in the array. There are several methods used to determine the right position where the current item can be inserted, all with varying degrees of efficiency. Three of those methods will now be discussed.

DETAILED EXAMPLE.

Consider an unsorted array of integers consisting of the following...

28, 16, 4, 12, 19, 11

In its simplest form, the insertion sort starts with the second item (16), and places it into a temporary location elsewhere. It then ranks it with preceding items, until either:

- (1) an item of lower or equal value is encountered, or
- (2) it runs out of items to compare, that is, the first item in the array is the current item.

As comparisons are being made, the items are shifted one position further down the array (that is, to the right in this example). When one of the above conditions is met, the current item is inserted into the vacant position created by the shifting of the last item compared. This is what is known as the straight insertion sort. Therefore, the following takes place:

16 is moved out to temp.

This is compared with 28, which is not \leq or = 16, so

28 is moved down one (to overwrite 16).

Then an attempt is made to compare 16 with the item before 28, which does not exist, so 16 is inserted at position 1 from its temporary storage location.

The next pass then considers the third item (4), and ranks this with the preceding 2 items, which are already sorted, and finds itself being placed in position 1 with the preceding items moved down. The following is a trace of the third pass:

<u>CURRENT ITEM</u>	<u>RANKED WITH</u>	<u>SHIFT?</u>
12, Moved to temp.		
12	28	28 to position 4
12	16	16 to position 3
12	4	4 not shifted
12 Insertion at position 2		

After which the array will look like...

4 12, 16, 28, 19, 11

Whereby the first 4 items are now in the correct sequence. This process continues until the last item (11) has been inserted into its correct position using the above algorithm. The following subroutine performs this task. Firstly, the data variables are:

N=array size, L=lower bound of array (=1)

A[]=array of items to be sorted

Over=flag for detecting end of pass.

```

390 I=2: TRUE=-1: FALSE=0
400 IF I>N THEN470
410 TEMP=A(I): J=I: OVER=FALSE
420 J=J-1
430 IF J<L
      A(L)=TEMP: OVER=TRUE
      ELSE
        IF TEMP<A(J)
          A(J+1)=A(J)
        ELSE
          A(J+1)=TEMP: OVER=TRUE
440 IF NOT OVER THEN420
450 IFPOINT(127,0)RESET(127,0)ELSESET(127,0)
460 I=I+1: GOTO400
470 RETURN

```

An additional feature of toggling a graphic point in the top right hand corner of the screen is included at the end of each pass. When the program is run, you will notice that this flashes randomly, indicating random pass sizes, compared with passes in the bubble sort becoming successively shorter. Notice the structure of the nested IF.THEN.ELSE construct in line 430 and its resemblance to that construct in the flowchart in figure 1. This makes for program readability and ease of understanding without becoming bogged down in unravelling the meaning of the structure. Notice also the use of the convenient boolean-simulation feature of level 2 BASIC, in the variable "Over" to indicate pass completion. This is possible because, in BASIC, 0=False; Anything else=True but only NOT -1 = 0. Hence -1 being chosen for true. So, if Over = -1 (true), NOT Over = 0 (false). These make for program readability, which becomes important in achieving easily maintained and self-documenting programs. (Plug for structured programming!) Now, back to the problem at hand. In order to utilize this subroutine, the following driver is proved. Note that you will have to insert a "GOTO 550" at the start of the subroutine in order to pass control to the driver when "RUN"ning or simply enter "RUN 550", for the purposes of this demo.

```

550 INPUT"NO. OF ITEMS";N: IF N < 1 THEN550
560 DIM A(N): RANDOM
570 FOR I=1TON
580   A(I)=RND(1000):PRINTA(I);
590 NEXT I
600 PRINT:INPUT"HIT ENTER TO START";I
610 L=1:GOSUB390 : ' CALL SORT SUBROUTINE
620 FOR I=1TON
630   PRINTA(I);
640 NEXT I

```

Figure 1 shows the flowchart for this sorting algorithm, "straight insertion" sorting.

ANALYSIS.

Straight insertion sorting requires that an average of N^2 comparisons and moves be made. This happens to be better than the average number in bubble sort. In bubble sort, each successive pass is reducing in size by 1, but all the items in the current pass size must be examined. However, in straight insertion sort, although the number of passes is approximately the same as bubble (increasing in size instead of decreasing), the average number of comparisons per pass is approximately half, because the probability of finding an item that is \leq the current item in a given pass before the half-way point is reached is 1 in 2. This accounts for the increase in efficiency of the straight insertion sort over the bubble sort (1.2 times on empirical evidence), and the random pass sizes indicated by the flashing graphic block.

A REFINEMENT.

One way to increase the efficiency of the straight insertion sort is to use multiple insertion. This involves picking up more than one item as a candidate for insertion, say 3 items. These are themselves sorted into (descending) order in a temporary storage area (a sort of a sort within a sort!) and then the highest value is ranked with the preceding items, moving them down if $>$ the candidate, except in our example, the preceding item is moved down 3 places (to take up the positions vacated by the 3 candidates), or more precisely, moved down N places, where N is = the No. of candidates remaining for insertion. Then, the next candidate is ranked, and

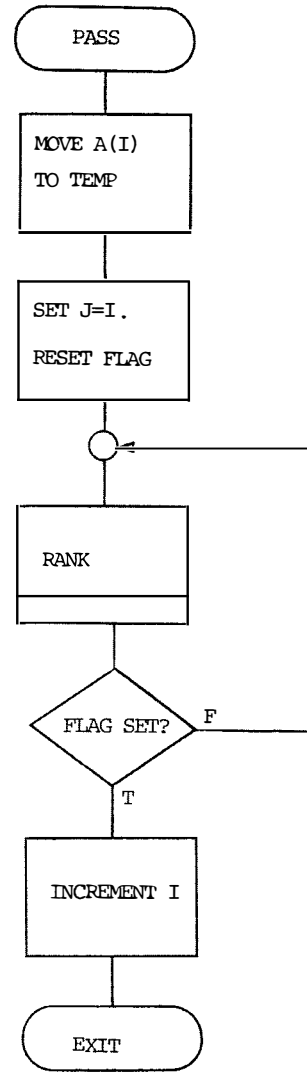
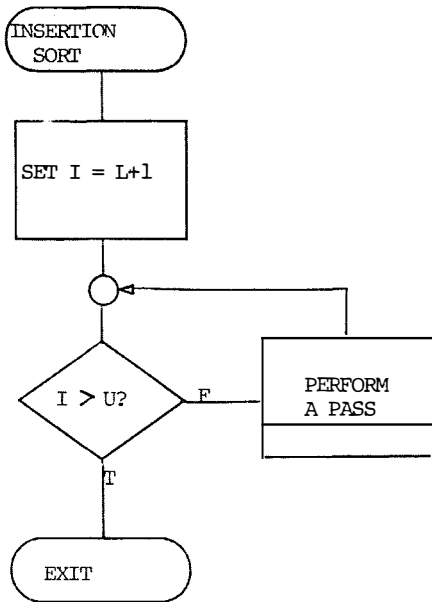


FIGURE 1 - INSERTION SORT ALGORITHM.

A = ARRAY BEING SORTED

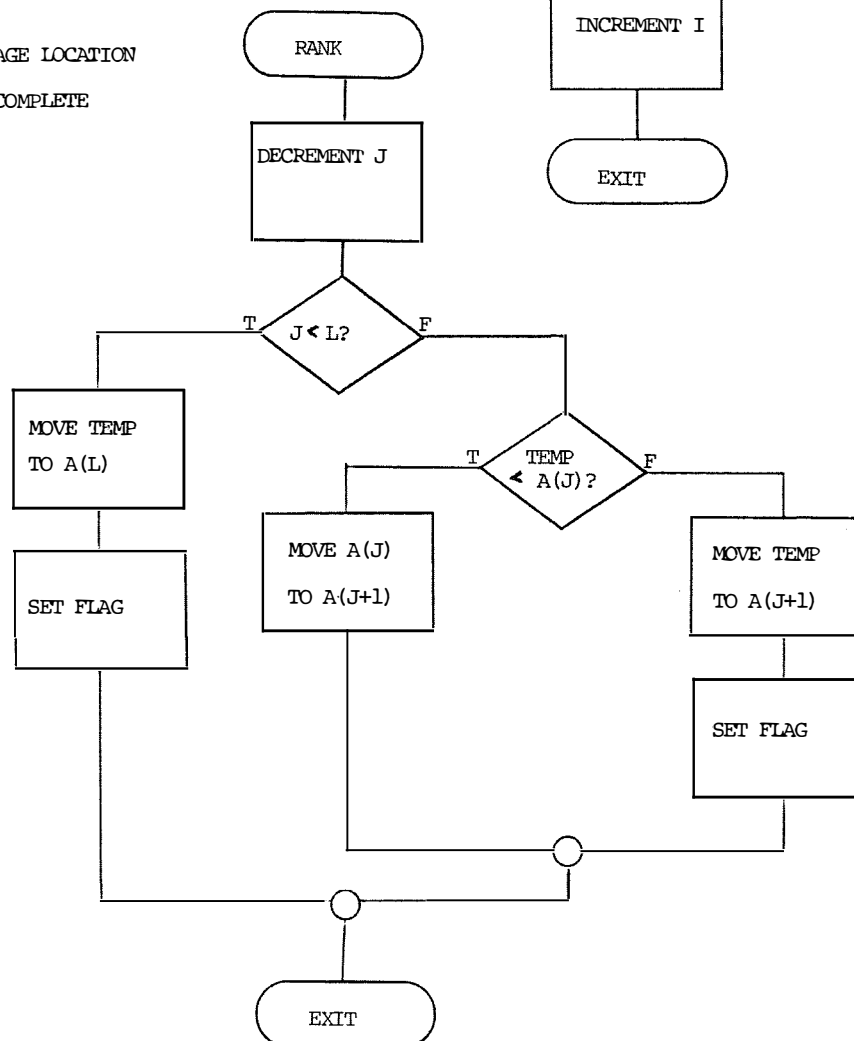
L = LOWER ARRAY BOUND (STARTING AT 1)

U = UPPER ARRAY BOUND

I, J = ARRAY INDICES

TEMP = TEMPORARY STORAGE LOCATION

FLAG SET WHEN PASS COMPLETE



if the ranked item must be moved down, it is moved 2 places, because there are 2 candidates left. You can get as complicated as you like by choosing more than 3 candidates, but remember that eventually an optimum point will be reached, and a trade-off must be made (referred to last article) between time spent in ranking, and time spent in manipulating the huge array of candidates, probably requiring a sophisticated sorting algorithm itself! - which obviously helps one to lose sight of the original problem.

BINARY SEARCHING.

Another way to increase the efficiency of the insertion sort is to develop a more efficient searching algorithm than sequential searching. Since the items being searched (compared with current key) are themselves in sorted order, we can make use of a semi-random searching technique that utilizes sorted codes, known as "binary searching". All that needs to be done is to tell it the lower and upper bounds of the array to be searched and the value of the current item for which a place is being searched. Binary searching a list involves the selection of the middle item and then comparing this with the item being searched for, adjusting the left or right bound depending on whether the middle item is $<$ or $>$ the item being searched. This "zeroing in" process continues until the left and right bounds overlap, or the item is found. You undergo a similar process when searching for an unfamiliar name in the telephone book, for example. The maximum number of inspections of a binary search is equal to: $(\text{Int}(\log \text{ of } N \text{ to Base } 2) + 1)$, or $(\log \text{ of } N \text{ to base } 2)$ if N is an exact multiple of 2, where N = total number of items in the search list.

For example, if list size (N) = 32, maximum number of inspections required to locate any item = 5, since 32 is an exact multiple of 2, ($2^5 = 32$), but if $N = 33$, then the maximum number of inspections = 6, according to the above formula. The average number must therefore be less than these figures. Compare that with the maximum number of inspections in a sequential search = 32, and an average = 16. If a binary search is employed to find the position where the current item is to be inserted, then the maximum number of comparisons to be made to find that position will be $(N-1) * (\text{Int}(\log N \text{ to base } 2) + 1)$, since there are $(N-1)$ passes. This represents a considerable increase in efficiency over the $N^2/4$ figure for straight insertion searching when N becomes larger. For example:

N	NUMBER OF SEARCHES	
	SEQUENTIAL SEARCHING	BINARY SEARCHING
5	7	12
10	25	36
20	100	95
50	625	294
100	2500	693

It is obvious that binary searching will be faster in locating the insertion position when N becomes approximately 15-20. And remember, the binary searching figures are the maximum that apply. In the average case, there will therefore be less searches than that indicated in the chart.

A SLIGHT DRAWBACK.

However, once the insertion position has been located using binary searching, it becomes necessary to move a block of items down one. This is an integral part of the straight insertion method, but needs to be implemented separately if insertion position is binary searched. This will reduce the efficiency of the binary insertion method somewhat but if implemented using the machine's block move instruction (e.g. LDIR, LDDR of the Z-80), then binary insertion will still be quicker than straight insertion by a considerable factor.

DEMONSTRATION PROGRAM

The following represents a BASIC version of a binary search routine, with driver:

```

1000 'PS=POSITION OF ITEM (-1 = NOT FOUND)
1010 'L,U=LOWER & UPPER BOUNDS OF CURRENT SEARCH LIST
1020 'MP=MIDPOINT OF LIST DELIMITED BY L & U
1030 'IV=INPUT VALUE BEING SEARCHED FOR
1040 '
1050 PRINT:INPUT"VALUE TO BE SEARCHED FOR";IV
1055 L=1:U=N
1060 GOSUB 1090
1070 IF PS=-1 THEN PRINT"NOT FOUND" ELSE PRINT"FOUND AT POSITION";PS
1075 GOTO1050
1080 ' SUBROUTINE BINARY SEARCH
1090 PS=-1
1100 IF L>U OR PS<>-1 THEN RETURN
1110 MP=INT((L+U)/2)
1120 IF IV<A(MP)
      U=MP-1

```

```

ELSE
  IF IV>A(MP)
    L=MP+1
  ELSE
    PS=MP
1130 GOTO1100
1140 END
    
```

This algorithm is formally defined in figure 2.

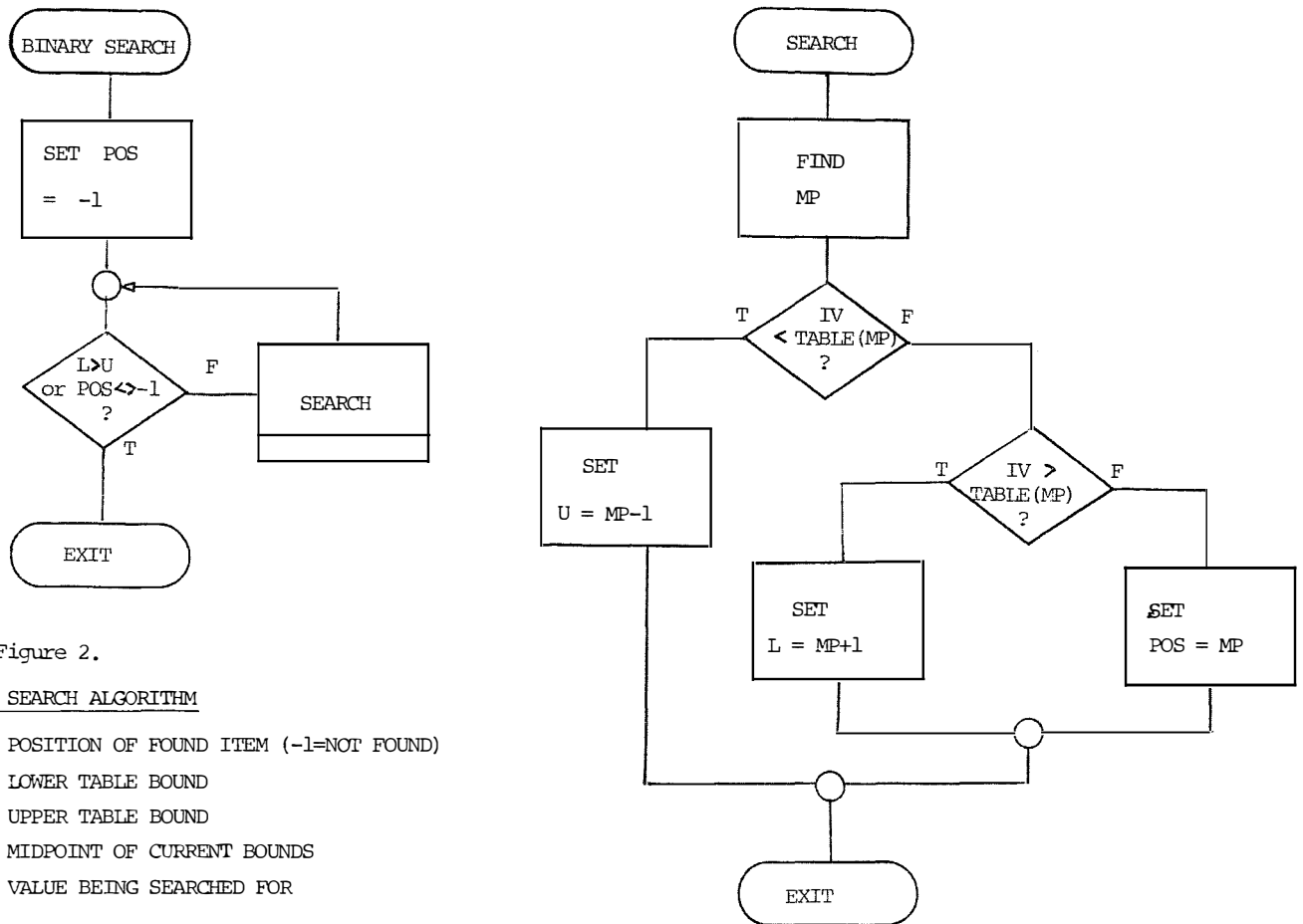


Figure 2.

BINARY SEARCH ALGORITHM

- POS = POSITION OF FOUND ITEM (-1=NOT FOUND)
- L = LOWER TABLE BOUND
- U = UPPER TABLE BOUND
- MP = MIDPOINT OF CURRENT BOUNDS
- IV = VALUE BEING SEARCHED FOR

Now, to make use of this code, as well as for purposes of demonstration, a sequenced search list must be supplied. Wonder where we can get one of those from? You guessed it. We can use the insertion sort above. If the binary search routine and driver have been keyed in to follow the insertion sort driver, then control will automatically fall through to the binary search driver, requesting item to be searched for.

A RECURSIVE APPROACH.

As an interesting sideline, the binary search is also conveniently defined recursively (that is, in terms of itself), and since BASIC does not support recursive functions or procedures, the following is a recursive definition of that same binary search, in Pascal. Those who own a Pascal compiler should be interested in the following code.

Assume the following declarations in the main program block:

```

TYPE LIST:ARRAY(1..N) OF REAL;
VAR TABLE:LIST; L,U,POSITION:INTEGER; VALUE:REAL;
:
:
PROCEDURE BINARYSEARCH (VAR TABLE:LIST; LOW,UPP:INTEGER;
VALUE:REAL; VAR POS:INTEGER);
VAR MID:INTEGER;
BEGIN
    
```

```

IF LOW>UPP THEN POS:= -1
ELSE
  BEGIN
    MID:= (LOW+UPP) DIV 2;
    IF VALUE<TABLE(MID) THEN
      BINARYSEARCH (TABLE,LOW,MID-1,VALUE,POS)
    ELSE
      IF VALUE>TABLE(MID) THEN
        BINARYSEARCH (TABLE,MID+1,UPP,VALUE,POS)
      ELSE
        POS:= MID
  END
END;

```

The remainder of the binary insertion sort could also be defined in Pascal, once a block move procedure is defined:

```

PROCEDURE INSERTSORT (VAR TABLE:LIST; FIRST, LAST:INTEGER);
(* ADDITIONAL DECLARATIONS *)
VAR I:INTEGER; TEMP:REAL;
  PROCEDURE BINARYSEARCH (DEFINED ABOVE)
    :
    :
  PROCEDURE BLOCKMOVE (VAR TABLE:LIST; L,U:INT);
  VAR I:INTEGER;
  BEGIN
    FOR I:=U DOWNT0 L DO TABLE(I+1):=TABLE(I)
  END;
BEGIN (* OF MAIN *)
  FOR I:=2 TO N DO
  BEGIN
    IF TABLE(I) < TABLE(I-1) THEN
      BEGIN
        TEMP:=TABLE(I);
        BINARYSEARCH (TABLE,1,I-1,TEMP,POSITION);
        BLOCKMOVE (TABLE,U+1,I-1);
        TABLE(U+1):= TEMP
      END
    END
  END;
END;

```

One of the advantages of Pascal is the ability to abstract and modularize the principals of an algorithm, which makes for problem orientation rather than machine orientation. This fact is demonstrated in the explanation of the binary insertion sort. The main block procedure statements of this program clearly show what is involved in a binary insertion sort:

1. Save current item to temp.
2. Search for insertion position for item currently in temp.
3. Move block down starting at insertion position, which has been determined by a value passed back from the binary search routine.
4. Put saved item into vacated position.

TO SUMMARIZE...

Another method of sorting is available, that of sorting by "insertion". There are several variations of such a method, all of which are quicker on the average than the bubble sort. They are:

```

  Straight insertion sort
  Multiple insertion sort
  Binary insertion sort.

```

Also, the binary search algorithm, as used in the binary insertion sort was presented, demonstrating one application of the use of sorting routines.

NEXT MONTH.

So far, sorting algorithms generally of efficiency to the order of N^2 have been discussed. Next month, the insertion saga continues, but by using a different kind of data structure than the simple array, and by employing another variation to the insertion sort, the efficiency can be improved to the order of $N * (\log N \text{ to Base } 2)$.

★ ★ SOFTWARE SECTION ★ ★

***** MEASUREMENTS L2/16K by G. Skoryk *****

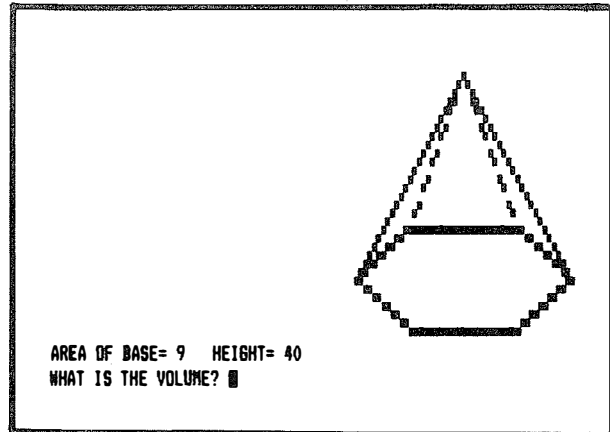
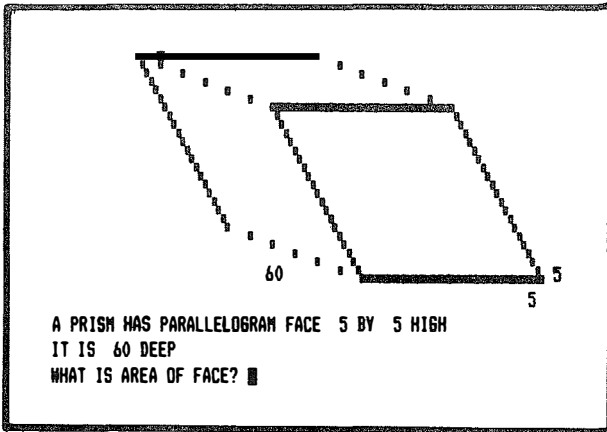
This program was developed to assist a high school student with her studies. It draws partially dimensioned figures on the screen and asks questions concerning the missing dimensions. It deals with the following figures:

- Cube
- Cylinder
- Cone
- Pyramid
- Sphere
- Rhombus Face Prism
- Parallelogram Face Prism
- Rectangle Face Prism
- Trapezium Face Prism
- Triangle Face Prism

There are two modes of operation:

- Check homework - user selects dimensions.
- Practice - computer selects dimensions at random.

The figure selected is drawn on the screen (to scale within the limits of the screen) and the operator is asked to answer questions. If the answer is wrong, the correct answer and method of calculation is displayed. At the completion of each set of problems, the progress score is displayed with the option to continue or quit.



SAMPLE SCREEN DISPLAYS

The program logic is as follows:

LINE NOS.

- 30-60 Assign string variables for repetitive words in text, thus saving a considerable amount of typing time.
- 190 Displays menu and allows operator to make selections.
- 110-130) Branch off program to sections according to selections made.
- 290 & 300)
- 310, 330 Subroutines which derive random numerical values for assignments.
- & 340
- 350-440 Subroutines which allows operator to nominate values.
- 450 Subroutine which calculates the area of face, whole surface area and volume of cube.
- 460, 470 Subroutines which print calculations and correct answers for cube if operator's
- & 480 answer is incorrect.
- 490-750 Subroutines which carry out similar operations for other figures.

- 760-900 Subroutines which describe the assignment and direct the program to other subroutines for processing.
- 910-1040 Subroutines which ask questions, compare operator's answers with correct answers and branch off program to other subroutines for further processing.
- Remainder Subroutines which provide data for graphics and print the graphics.

***** SUPER HANGMAN L2/16K by Martin Downey *****

Super Hangman is based on the original word game of the same name but with a difference. It provides continuous, animated graphics, music and other sound effects, two levels of play and has a 418 word vocabulary built in which can be expanded still further in 32K and 48K machines.

SUPER HANGMAN is a Level II BASIC program and so is loaded using the CLOAD command. After it has loaded successfully, type in RUN and press ENTER (or NEW LINE). If you want SOUND merely connect an amplifier and speaker to the cassette output plug (normally plugs into the MIC socket). If two cassette ports are available either one can be used.

After ENTERing RUN a title page will be displayed. Then the BANDIT will appear accompanied by a Mexican tune. You can begin the game straight away by pressing B or go through the instructions by pressing I or just waiting for the tune to finish. Before each game you must press J or S to indicate Junior or Senior level. After this, the mystery word is chosen by the computer and the number of letters is shown at the bottom of the screen (each - represents a letter).

The game is played along the lines of the original HANGMAN games. You must guess the mystery word by guessing one letter at a time, or the whole word at once. For each letter guess, press the appropriate key. The letter will be displayed and noticed by the BANDIT. If the letter occurs in the mystery word it will be inserted in ALL the places that it occurs. If the letter doesn't occur in the word, it will be displayed on the left of the screen and the GALLOWS will be added to. Six wrong guesses (letters and words) and you will lose. If you guess all the letters or the whole word then you will win. If you try a letter guess more than once, you will be told accordingly but not penalised. You will be warned when you are on your last guess.

To guess the whole word in one go, first press the @ key. You will then be told to type in your guess and press ENTER (NEW LINE) when you are finished. The computer will ignore all blanks (spaces) even if inside the word (i.e. DOG, D O G and DO G are all seen the same).

At the end of a game you can play again by pressing the Y key or end by answering with the N key. Although SOUND is not essential for the enjoyment of SUPER HANGMAN it is heartily recommended. (There are 5 tunes and numerous sound effects). The 418 word vocabulary should be enough for even the most regular player. However, if you have more than 16K memory, you might like to extend it. Simply add the extra words at the end of the program and change the value in line 2120 from 418.

***** FILES L2/48K by E. Paay modified by N.E.L. Rossiter *****

This program was originally published in Issue 3, February, 1980. We have had a number of requests from readers for a 48K version. Mr. Rossiter has now modified it and has made some useful improvements as well. With the original program, it was necessary to set Memory Size, load in a machine language routine and then load the BASIC Program. With this new version, the one BASIC program carries out the whole job by POKEing the machine language routine into memory. The Listing has not been renumbered to assist those who have already keyed in the original program, to make the necessary changes.

Files is designed for Level 2 systems. It allows the user to enter data into a file, to recall this data later and list it to the screen, in a format set out by the user. It is therefore suitable for such functions as a mailing list containing names, addresses and phone numbers or a directory to contain, for instance, magazine articles, date, volume number and type of article, etc. The program is then capable of searching the file for any data asked for by the user and displaying all files to the screen which contain the data asked for.

Files overcomes many problems associated with this kind of program. It avoids lengthy, time-consuming data transfers from tape, as would be the case if the INPUT#-1 command is used to load and write the data to tape separately. With this system all data stays with the program at all times as data statements. It is therefore only necessary to CLOAD or CSAVE once to load the program and data from tape (or disk).

TYPING THE PROGRAM IN:

As was stated earlier, this is a part machine language and part BASIC program. Now examine the listing of the program and see if you can find the machine language subroutine..

Have you guessed it? It is the strange looking numbers and letters in line 2 and 3. That is why it appears to make no sense. This m/l subroutine allows the user to type in data without having to stop the program and type in complete data lines manually.

The m/l subroutine creates data lines, numbers them, and changes the necessary memory pointers for BASIC so that program execution does not have to be interrupted. It also saves a lot of time and effort for the user. To type this part of the program in, type in the lines 2 and 3 as they appear in the listing in the magazine. Type in the rest of the program which is in BASIC. The only other thing to be mentioned is that the text lines such as lines 2 and 3 must be typed in exactly as listed.

Also note that the data statements at the end of the program don't have to be typed in. They are there for demonstration purposes, and show how data is stored. They can be typed in however so that the user can practise using the program. When you have finished typing in the program, CSAVE it (disk users can SAVE it).

THE COMMANDS:

(1) Enter new data to file.

This command allows the user to add data to the file. All that is required is for the user to type in the data in the correct order separated with comma's. Because all data is stored as strings, the data must be stored between double quotes. It is not necessary however for the user to type these quotes in manually because as soon as a comma is typed, it is immediately converted to: ",", automatically.

If data has been typed in incorrectly, BASIC will return with an SN error in the data line concerned when it reads the data later - so take care!

When a complete "set" or "group" of data has been typed in, type "ENTER" then when the "READY" message appears, type ENTER again. You will then be able to enter the next set of data or hit the DOWN ARROW/ESC key to return to the directory.

Also, you will notice that the amount of free memory left is displayed on the screen so that you will know when you come to the end of your memory.

DATA FORMAT:

The first set of data you type in will be used as headings for all other data. Examine the first data statement in the listing of the program to see what I mean. (Or better still, type in data as in listing and try).

The format of the first set of data is different from all others. It contains a number at the beginning of the data statement. The number tells the program how many columns of data there are in each group of data.

Following this number are the sub-headings, (i.e. if the number is 3, then it must be followed by 3 sub-headings in the first line and all other data groups are expected to have 3 lots of data in its group or set).

If you look at the listed data lines you will notice an "@" followed by a number. The "@" tells the program that the data following contains the line number. This is not typed in by the user. It is done automatically when you enter data under the "ENTER DATA" command.

(2) Search Command.

This command will search the file for the data asked for by the user. It will then list the first occurrence of the data concerned. Answer the question "CONT. OR STOP" with "C" to search the rest of the file, or with "S" to stop. Note also that it is often not necessary to type in the complete word or sentence you are searching for. For instance, if your file contains the words "bicycle" and "motorcycle" then the search command will list both if it is told to search for "cycle". The search routine will tell you if data is not found or if all data has been searched.

(3) Exit from program.

This command can be used to stop the program. Under Lev 2 the BREAK key can also be used to stop execution. Under Disk BASIC, however, the exit command must always be used to stop the program and not the BREAK key or Disk I/O will not function properly.

(4) Edit and List line numbers.

This command tells you how to edit the data lines but does not actually perform the edit function itself. To edit a file, stop the program and use the edit function provided by BASIC. (Remember, don't edit line 1).

This command can, however, search for a particular string or data in its files and list all line numbers of data lines that contain the string in question so that the user will know which lines to edit.

(5) List All.

This command will list all data to the screen, one file at a time. Type "Y" to continue listing and "N" to stop, when asked.

***** DATA BASE MANAGEMENT SYSTEM 48K Disk by G. Moad *****

This program makes use of the NEWDOS 80 ver 2. CMD"0" sort routine. For those people who do not have Version 2, the author has provided the necessary information to make the program run using the Tandy sort routine published in the Microcomputer news of May 1981.

The database management system is for a 48K, 1 (or more) disk TRS-80. (The program could be used on a 32K machine, however, only a very limited database could be handled).

To run the program:

- (1) Type in the program from the magazine.
- (2) Save it to disk, with the filename DBMS/BAS
- (3) Put the disk in drive 0
- (4) Press reset : boot the system
- (5) Type " BASIC,40960 : enter basic with memory size set at 40960
- (6) Type : RUN "DBMS/BAS": run the program
- (7) Then follow the instructions given by the program.

This program has advantages over some commercial database programs I have seen.

- (1) It can be readily modified (by me at least!)
- (2) A fast machine language sort is used (NEWDOS/80 Version 2's CMD"0")
- (3) Text is stored in protected memory so as to be immune from BASIC's rearranging of string space
- (4) Lower case is permitted within fields without a lower case driver being necessary
- (5) The number of records the system can hold is determined by the program taking into account the length of the fields used.

This program was written to work with NEWDOS/80 version 2. However, it will also work with TRSDOS 2.3 if the sort routine is replaced.

Some use has been made of the following routine which enables strings to be located anywhere in memory (e.g. in screen memory or in protected memory). This has a number of uses:

- (1) it simplifies some display routines
- (2) it makes it possible to read the screen into a string
- (3) it puts strings out of reach of BASIC's attempts to move them around
- (4) it gives a means of accessing parts of a string which is often simpler than using LEFT\$, RIGHT\$, or MID\$

Note, however, that one is limited in the ways of transferring data into these strings. The commands LSET, RSET, and POKE can all be used. However, statements of the form S\$=expression cannot be used. If the string is located in screen memory one can simply write to it using PRINT @ or if it is in protected memory, it is possible to use MID\$ on the left hand side of an expression. The following short program gives a trivial, but annotated, example of its use (it is on the disk as TESTSTR/BAS).

```

10 CLS
20 S$="" 'STRING MUST BE INITIALIZED
30 ML=15360 'MEMORY LOCATION
40 LN=64 'LENGTH OF STRING
50 POKE VARPTR(S$)+2, INT(ML/256) 'POKE MSB
60 POKE VARPTR(S$)+1, ML-INT(ML/256)*256 'POKE LSB
70 POKE VARPTR(S$), LN 'POKE LENGTH OF STRING
80 RSET S$ = "TEST" 'PUT TEST AT TOP RIGHT OF SCREEN
90 FOR I = 1 TO 100 : NEXT 'DELAY
100 LSET S$ = "TEST" 'PUT TEST AT TOP LEFT OF SCREEN
110 FOR I=1 TO 100 : NEXT : GOTO 80 'DELAY THEN DO IT AGAIN

```

Since there may be some readers who do not have access to NEWDOS/80 version 2, the following modifications can be made to the program to permit use of the sort routine published in Tandy's TRS-80 Microcomputer News of May 1981. Note that with these modifications the program only allows a sort by field number 1 - multiple sort keys are not supported. Alternatively, readers may wish to use their own sort routine. This can be done simply by replacing the subroutine at line 350.

NOTE: TRSDOS users should load SORT/CIM before entering BASIC.

```
25 DIMX(2):CMD"LOAD SORT/CIM":DEFUSR=&HFF00
350 Z=0:X(0)=NR:X(1)=VARPTR(AR(0)):Z=USR(VARPTR(X(0))):RETURN
```

NOTE : Line 25 is INSERTED and Line 350 REPLACES the existing line.

The following functions are available:-

ADD - New records are appended to the end of the file. When an entire record has been entered an opportunity to EDIT or DELETE the record just entered is given.

DELETE - After entering the delete command an opportunity is always given for you to change your mind before the deletion is carried out. The file is sorted according to the last used or the default sort key following DELETE.

EDIT - To edit a field entry it is necessary to re-enter the entire field. To clear a field enter a space. To leave a field unchanged, just press enter.

FIND - Enter either the number or the name of the field to be searched. Enter enough characters to identify the expression to be found. All records in which the specified field BEGINS with the entered string will be found.

FIND2 - To find records which contain the expression WITHIN the field preface the string to be found with a '*'. All records in which the specified field CONTAINS the string (-'*) will then be found.

FIND3 - Long form format displays entire records with field names and gives the opportunity to EDIT or DELETE the record. Short form format displays one record per line, ten records per page. The records appear in an abbreviated format.

FIND4 - Note that if the N(ext) or P(revious) keys are used during the long format search the system will continue its search from the current (displayed) record when the F(ind)key is pressed.

INTEGER FIELDS - Integers or fixed point numbers which vary in length will be sorted correctly only if they are placed in integer fields. Floating point numbers may not be sorted correctly by the program.

LIST - Records are listed one at a time with field names. An opportunity to EDIT or DELETE the record is given.

MERGE - A file from disk is merged with the resident file. The file to be MERGED must have the same number of fields, and the fields must be of the same length and type, as those of the resident file.

NEW - The resident file is cleared from memory and the program is reinitialized.

PRINT - The file is printed to the video or to the video and the line printer according to the specified parameters. The Q(uit)key may be used to abort printing.

PRINT2 - The print routine was designed to work with a TANDY Line Printer VIII. Its correct operation with other printers may require some changes. For example, with the LPVII LPRINT will cause a line feed (Line 1435 of program).

SORT - The entire file is sorted in ascending order according to the specified sort key. The sort will be completed most rapidly if as few as possible fields are included in the sort key. This program uses the NEWDOS/80 Version 2 sort.

WRITE - The file is written to disk in 2 parts. 'NAME'/DAT contains data describing the nature of the file. 'NAME'/TXT contains the text.

***** DUPLEX L2/m.1 by N. Rossiter *****

DUPLEX is a package of double precision mathematical functions which is called by a special use of the USR function; since it disables this function in its normal use, DUPLEX also provides a means of accessing the user's own machine language routines, using the DEF command to define entry points. It loads at the bottom of user memory, from 42E9H to 4610H, and sets the BASIC pointers past itself to 4611H. It should therefore be loaded first, before loading or entering your BASIC program. It is not intended for use with Disk BASIC.

**DON'T BE HELD BACK BY AN
ANTIQUATED DISK OPERATING SYSTEM
MOVE UP TO**

NEWDOS 80 **\$149 incl. p&p**

NEWDOS 80 is a completely new DOS for the TRS-80 SYSTEM 80. It is well-documented, bug free and increases the power of your system many times over. It is upward compatible with TRSDOS AND NEWDOS (ie TRSDOS and NEWDOS+ programs will run on NEWDOS 80 but the reverse is not necessarily so).

These are just a few of the many new features offered by NEWDOS 80.

- * New BASIC commands that support variable record lengths up to 4095 bytes long.
- * Mix or match disk drives. Supports any track count from 18 to 96. Use 35, 40, 77 or 80 track 5¼ inch mini disk drives, 8 inch disk drives OR ANY COMBINATION.
- * An optional security boot-up for BASIC or machine code application programs. User never sees "DOS-READY" or "READY" and is unable to "BREAK", clear screen or issue any direct BASIC statements, including "LIST".
- * New editing commands that allow program lines to be deleted from one location and moved to another or to allow the duplication of a program line with the deletion of the original.
- * Enhanced and improved RENUMBER that allows relocation of subroutines.
- * Create powerful chain command files which will control the operation of your system.
- * Device handling for routing to display and printer simultaneously.
- * MINIDOS — striking the D, F and G keys simultaneously calls up a MINIDOS which allows you to perform many of the DOS commands without disturbing the resident program.
- * Includes Superzap 3.0 which enables you to display/print/modify any byte in memory or on disk.
- * Also includes the following utilities:
 - Disk Editor/Assembler
 - Disassembler (Z80 machine code)
 - LM offset — allows transfers of any system tape to Disk file — automatically relocated.
 - LEVEL 1 — Lets you convert your computer back to Level 1.
 - LVIDKSL — Saves and loads Level 1 programs to disk.
 - DIRCHECK — Tests disk directories for errors and lists them.
 - ASPOOL — An automatic spooler which routes a disk file to the printer whilst the computer continues to operate on other programs.
 - LCDVR — a lower case drives which display lower case on the screen if you have fitted a simple lower case modification.

**DISK DRIVE USERS
ELIMINATE CRC ERRORS
AND
TRACK LOCKED OUT MESSAGES
FIT A PERCOM DATA SEPARATOR
\$37.00 plus \$1.20 p&p.**

When Tandy designed the TRS-80 expansion interface, they did not include a data separator in the disk-controller circuitry, despite the I.C. manufacturer's recommendations to do so. The result is that many disk drive owners suffer a lot of Disk I/O errors. The answer is a data separator. This unit fits inside your expansion interface. It is supplied with full instructions and is a must for the serious disk user.

**MPI DISK DRIVES
HIGHER PERFORMANCE — LOWER PRICE**

MPI is the second largest manufacturer of disk drives in the world. MPI drives use the same form of head control as 8" drives and consequently, they have the fastest track-to-track access time available — 5msec! All MPI drives are capable of single or double-density operation. Double-density operation requires the installation of a PERCOM doubler board in the expansion interface.

As well as single head drives, MPI also makes dual-head drives. A dual-head drive is almost as versatile as two single-head drives but is much cheaper.

Our MPI drives are supplied bare or in a metal cabinet — set up to operate with your TRS-80 or SYSTEM 80. All drives are sold with a 90 day warranty and service is available through MICRO-80 PRODUCTS.

MPI B51 40 Track Single Head Drive.only \$349
MPI B52 40 Track Double Head Drive.only \$449

Prices are for bare drives and include p&p. Add \$10.00 per drive for a cabinet and \$60.00 for a power supply to suit two drives. 40 track drives are entirely compatible with 35 track drives. A 40 track DOS such as NEWDOS 80 is necessary to utilise the extra 5 tracks.

**OVER 800 KILOBYTES ON ONE DISKETTE!
WITH MPI 80 TRACK DRIVES**

MPI 80 track drives are now available. The B91 80 track single-head drive stores 204 Kilobytes of formatted data on one side of a 5¼ inch diskette in single-density mode. In double-density mode it stores 408 Kilobytes and loads/saves data twice as quickly.

The B92 80 track dual-head drive stores 204 Kilobytes of formatted data on EACH side of a 5¼ inch diskette in single-density mode. That's 408 Kilobytes per diskette. In double-density mode, the B92 stores a mammoth 408 Kilobytes per side or 816 Kilobytes of formatted data per diskette. With two B92's and a PERCOM double, you could have over 1.6 Megabytes of on line storage for your TRS-80 for less than \$1500!!

MPI B91 80 Track Single Head Drive.only \$499
MPI B92 80 Track Dual Head Driveonly \$619

Prices are for bare drives and include p&p. Add \$10.00 per drive for a cabinet and \$60.00 for a power supply to suit two drives. Note: 80 track drives will not read diskettes written on a 35 or 40 track drive. If drives with different track counts are to be operated on the same system, NEWDOS 80 must be used.

**CARE FOR YOUR DISK DRIVES?
THEN USE
3M's DISK DRIVE HEAD CLEANING DISKETTES
\$30.20 incl. p&p.**

Disk drives are expensive and so are diskettes. As with any magnetic recording device, a disk drive works better and lasts longer if the head is cleaned regularly. In the past, the problem has been, how do you clean the head without pulling the mechanism apart and running the risk of damaging delicate parts. 3M's have come to our rescue with SCOTCH BRAND, non-abrasive, head cleaning diskettes which thoroughly clean the head in seconds. The cleaning action is less abrasive than an ordinary diskette and no residue is left behind. Each kit contains:

- 2 head cleaning diskettes
- 1 bottle of cleaning fluid
- 1 bottle dispenser cap

USE TANDY PERIPHERALS ON YOUR SYSTEM-80 VIA SYSPAND-80 - \$97.50 incl. p&p

The SYSTEM-80 hardware is not compatible with the TRS-80 in two important areas. The printer port is addressed differently and the expansion bus is entirely different. This means that SYSTEM-80 owners are denied the wealth of economical, high performance peripherals which have been developed for the TRS-80. Until now, that is. MICRO-80 has developed the SYSPAND-80 adaptor to overcome this problem. A completely self-contained unit in a small cabinet which matches the colour scheme of your computer, it connects to the 50-way expansion part on the rear of your SYSTEM 80 and generates the FULL Tandy 40 way bus as well as providing a Centronics parallel printer port. SYSPAND-80 enables you to run an Exatron Stringy Floppy from your SYSTEM 80, or an LNW Research expansion interface or any other desirable peripherals designed to interface to the TRS-80 expansion port. Make your SYSTEM 80 hardware compatible with the TRS-80 via SYSPAND-80.

PROGRAMS BY MICROSOFT

EDITOR ASSEMBLER PLUS (L2/16K) \$37.50 + \$1.20 p&p

A much improved editor-assembler and debug/monitor for L2/16K TRS-80 or SYSTEM 80. Assembles directly into memory, supports macros and conditional assembly, includes new commands-substitute, move, copy and extend.

LEVEL III BASIC \$59.95 plus \$1.20 p&p

Loads on top of Level II BASIC and gives advanced graphics, automatic renumbering, single stroke instructions (shift-key entries) keyboard debounce, suitable for L2/16K and up (Not Disk BASIC)

ADVENTURE ON DISK \$35.95 plus \$1.20 p&p

This is the original ADVENTURE game adapted for the TRS-80. The game fills an entire diskette. Endless variety and challenge as you seek to rise to the level of Grand Master. Until you gain skill, there are whole areas of the cave that you cannot enter. (Requires 32K One Disk)

BASIC COMPILER \$2.08 plus \$2.00 p&p

New improved version, the Basic Compiler converts Disk BASIC programs to machine code, automatically. A compiled program runs, on average, 3-10 times faster than the original BASIC program and is much more difficult to pirate.

UPGRADE TO 16K FOR ONLY \$30.00!!

MICRO-80's 16K MEMORY EXPANSION KIT HAS BEEN REDUCED IN PRICE EVEN MORE

Larger volume means we buy better and we pass the savings on to you. These are our proven, prime, branded 200 ns (yes, 200 nanosecond) chips. You will pay much more elsewhere for slow, 350 ns. chips. Ours are guaranteed for 12 months. A pair of DIP shunts is also required to upgrade the CPU memory in the TRS-80 - these cost an additional \$4.00. All kits come complete with full, step-by-step instructions which include labelled photographs. No soldering is required. You do not have to be an experienced electronic technician to instal them.

DISK DRIVE CABLES SUITABLE FOR ANY DISK DRIVES

DC-2 2 Drive Connector Cable \$39 incl. p&p
DC-4 4 Drive Connector Cable \$49 incl. p&p

DOUBLE THE SPEED AND CAPACITY OF YOUR DISK DRIVES PERCOM DOUBLER ONLY \$220 plus \$2.00 p&p

Installing a Doubler is like buying another set of disk drives, only much cheaper!! The doubler works with most modern disk drives including:- MPI, Micropolis, Pertec, TEAC (as supplied by Tandy). The doubler installs in the TRS-80 expansion interface, the System-80 expansion interface and the LNW Research expansion interface in a few minutes without any soldering, cutting of tracks, etc. It comes complete with its own TRSDOS compatible double density operating system.

DOUBLE-ZAP II - DOUBLE DENSITY PATCH FOR NEWDOS 80

ONLY \$53.00 plus \$1.00 p&p

If you are using NEWDOS 80, then you also need DOUBLE-ZAP II on diskette. This program upgrades your NEWDOS 80 to double density with ADR (automatic density recognition). It retains all the familiar features, including the ability to mix and match track counts on the same cable. In addition, it gives NEWDOS 80 the ability to mix densities on the same cable, automatically. If you place a single density diskette in drive 0, say and a double density diskette in drive 1, Double-ZapII will recognise this and read/write to drive 0 in single density whilst at the same time it reads/writes to drive 1 in double density!

FLOPPY DOCTOR AND MEMORY DIAGNOSTIC (by MICRO CLINIC) \$29.95 plus 50c. p&p

Two machine language programs on a diskette together with manual which thoroughly test your disk drives and memory. There are 19 possible error messages in the disk drive test and their likely causes are explained in the manual. Each pass of the memory tests checks every address in RAM 520 times, including the space normally occupied by the diagnostic program itself. When an error occurs the address, expected data, and actual data are printed out together with a detailed error analysis showing the failing bit or bits, the corresponding IC's and their location. This is the most thorough test routine available for TRS-80 disk users.

BOOKS

LEVEL II ROM REFERENCE MANUAL \$24.95 + \$1.20 p&p

Over 70 pages packed full of useful information and sample programs. Applies to both TRS-80 and SYSTEM 80.

TRS-80 DISK AND OTHER MYSTERIES \$24.95 + \$1.20 p&p

The hottest selling TRS-80 book in the U.S.A. Disk file structures revealed, DOS's compared and explained, how to recover lost files, how to rebuild crashed directories - this is a must for the serious Disk user and is a perfect companion to any of the NEWDOS's.

LEARNING LEVEL II \$16.95 + \$1.20 p&p

Written by Daniel Lien, the author of the TRS-80 Level I Handbook, this book teaches you, step-by-step, how to get the most from your Level II machine. Invaluable supplement to either the TRS-80 Level II Manual or the System-80 Manuals.

MORE AUSTRALIAN SOFTWARE

All programs designed to run on both the TRS-80 or the SYSTEM 80 without modification. Most programs include sound

TRIAD VOL 1 – L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Three separate games which test your powers of memory and concentration. The programs combine graphic displays and sound:

SIMON-SEZ: Just like the electronic music puzzles on sale for more than \$20. Numbers are flashed on the screen and sounded in a sequence determined by the computer. Your task is to reproduce the sequence, correctly.

LINE?: Rather like a super, complicated version of noughts and crosses. You may play against another player or against the computer itself. But beware, the computer cheats!

SUPER CONCENTRATION: Just like the card game but with more options. You must find the hidden pairs. You may play against other people, play against the computer, play on your own, or even let the '80 play on its own.

TRIAD VOL 2 – L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Remember those "NUMERO" puzzles in which you had a matrix of numbers (or letters) with one blank space and you had to shuffle the numbers around one at a time until you had made a particular pattern? Well, **SHUFFLEBOARD**, the first program in this triad, is just this, except that the computer counts the number of moves you take to match the pattern it has generated — so it is not possible to cheat.

MIMIC is just like SHUFFLEBOARD except that you only see the computer's pattern for a brief span at the beginning of the game, then you must remember it!

In **MATCHEM**, you have to manoeuvre 20 pegs from the centre of the screen to their respective holes in the top or bottom rows. Your score is determined by the time taken to select a peg, the route taken from the centre of the screen to the hole and your ability to direct the peg into the hole without hitting any other peg or the boundary.

VISURAMA L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Two programs which give fascinating, ever-changing patterns on the screen.

LIFE is the fastest implementation of the Game of Life you will see on your '80. Machine language routines create up to 1200 new generations per minute for small patterns or up to 100 per minute for the full 128 x 48 screen matrix. Features full horizontal and vertical wraparound.

EPICYCLES will fascinate you for hours. The ever-changing ever-moving patterns give a 3D effect and were inspired by the ancient Greek theories of Ptolemy and his model of the Solar system.

EDUCATION AND FUN – L1/4K, L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Written by a primary school teacher to make learning enjoyable for his pupils, there are five programs in both Level I and Level II to suit all systems:

BUG-A-LUG: a mathematics game, in which you must get the sum correct before you can move.

AUSTRALIAN GEOGRAPHY: learn about Australian States and towns, etc.

SUBTRACTION GAME: build a tower with correct answers.

HOW GOOD IS YOUR MATHS? Select the function (+, -, ÷ or X) and degree of difficulty.

HANGMAN: That well known word game now on your computer.

Recommended for children from 6 to 9 years.

COSMIC FIGHTER & SPACE JUNK – L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Both programs have sound to complement their excellent graphics. In **COSMIC FIGHTER**, you must defend the earth against seven different types of alien aircraft. It is unlikely that you will be successful but you will have a lot of fun trying!

Your mission in **SPACE JUNK** is to clean up all the debris left floating around in space by those other space games. It is not as simple as it sounds and space junk can be quite dangerous unless you are very careful.

SPACE DRIVE L2/4K & 16K Cassette \$8.95 Disk \$13.95 + 60c p&p

Try to manoeuvre your space ship through the meteor storms then land it carefully at the space port without running out of fuel or crashing. Complete with realistic graphics.

STARFIRE AND NOVA INVASION L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

Both programs include sound to improve their realism.

STARFIRE seats you in the cockpit of an X-wing fighter as you engage in battle with the deadly Darth Vader's Tie-fighters. Beware of the evil one himself and may the Force be with you.

In **NOVA INVASION**, you must protect your home planet of Hiberna from the invading NOVADIANS. You have two fixed guns at each side of the screen and a moveable one at the bottom. Apart from shooting down as many invaders as possible, you must protect your precious hoard of Vitaminium or perish!

AIR ATTACK AND NAG RACE – L2/16K Cassette \$10.95 Disk \$15.95 + 60c p&p

An unlikely combination of programs but they share the same author who has a keen sense of humour.

AIR ATTACK includes sound and realistic graphics. The aircraft even have rotating propellers! But they also drop bombs on you, so it's kill or be killed!

NAG RACE lets you pander to your gambling instinct without actually losing real money. Up to five punters can join in the fun. Each race results in a photo-finish whilst there is a visible race commentary at the bottom of the screen throughout the race. Happy punting!

FOUR LETTER MASTERMIND L2/16K Cassette \$8.95 Disk \$13.95 + 60c p&p

There are 550 four-letter words from which the computer can make its choice. You have 12 chances to enter the correct word. After each try, the computer informs you of the number of correct letters and those in the correct position. You can peek at the list of possible words but it will cost you points. Makes learning to spell fun.

MUSIC IV – L2/16K Cassette \$8.95 Disk \$13.95 + 60c p&p

Music IV is a music compiler for your '80. It allows you to compose or reproduce music with your computer that will surprise you with its range and quality. You have control over duration (full beat to 1/16 beat) with modifications to extend the duration by half or one third for triplets. Both sharps and flats are catered for as are rests. Notes on whole sections may be repeated. The program comes with sample data for a well-known tune to illustrate how it is done.

*** SAVE 00\$'s *** SAVE 00\$'s *** SAVE 00\$'s *** MICRO-80 EXPANSION INTERFACE ***

MICRO-80's expansion interface utilises the proven LNW Research Expansion board. It is supplied fully built up and tested in an attractive cabinet with a self contained power supply, ready to plug in and go. The expansion interface carries MICRO-80's full, no hassle, 90-day warranty.

Features include: ● Sockets for up to 32K of memory expansion ● Disk controller for up to 4 disk drives ● Parallel printer port ● Serial RS232C/20mA I/O port ● Second cassette (optional)

The expansion interface connects directly to your TRS-80 L2/16K keyboard or, via SYSPAND-80 to your SYSTEM-80 VIDEO GENIE
Prices: HD-010-A Expansion Interfaces with Ø K : \$499.00 HD-010-B Expansion Interfaces with 32K : \$549.00 HD-011 Data separator fitted (recommended) : add \$29.00 HD-012 Dual cassette Interfaces fitted : add \$19.00

The MICRO-80 Expansion Interface is also available in kit form.

Prices: HD-013 Kit consisting of LNW Research PC board and manual, ALL components including cabinet & power supply : \$375.00 HD-011 Data separator for above \$25.00 HD-013 Dual cassette Interface kit : \$15.00

**TURN
THIS**

**into
this**

for \$49.00 plus \$2.00 p & p

A choice of upper and lower case display is easier to read, gives greater versatility.

The Micro-80 lower case modification gives you this facility, plus the symbols for the 4 playing-card suits for \$49.00 + \$2.00 p. & p.

The Micro-80 modification features true below-the-line descenders and a block cursor.

Each kit comes with comprehensive fitting instructions and two universal lower-case drive routines on cassette to enable you to display lower case in BASIC programs.

The driver routines are self-relocating, self-protecting and will co-reside with other machine language programs such as Keyboard-debounce, serial interface driver programs etc.

Both programs give your TRS-80™ Model I or System 80™ an optional typewriter capability, i.e. shift for upper case.

The second programme also includes Keyboard-debounce and a flashing cursor.

You fit it. Or we can.

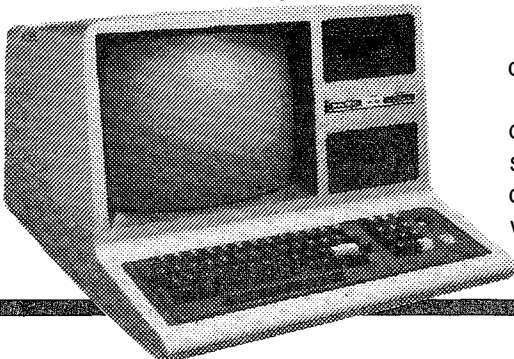
Fitting the modification requires soldering inside the computer. This should only be carried out by an experienced hobbyist or technician.

If you are at all dubious, a fitting service is available in all capital cities for only \$20.00.

A list of installers is included with each kit.

Save \$120 now.

ADD A DISK DRIVE TO YOUR TRS-80™ MODEL III FOR ONLY \$875.00 OR ADD TWO FOR ONLY \$1199.



The Micro-80 disk drive upgrade for the TRS-80™ Model III contains the following high quality components:

1 or 2 MPI 40-track single head disk drives, 1 VR Data double-density disk controller board and 1 dual drive power supply plus all the necessary mounting hardware, cables and comprehensive fitting instructions, which can be carried out with a minimum of fuss by any average computer owner.

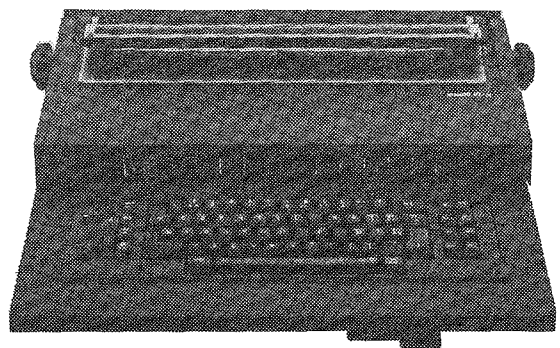
Fitting service is available for \$25.00 in most capital cities.

ONLY \$2049 INC. S.T.

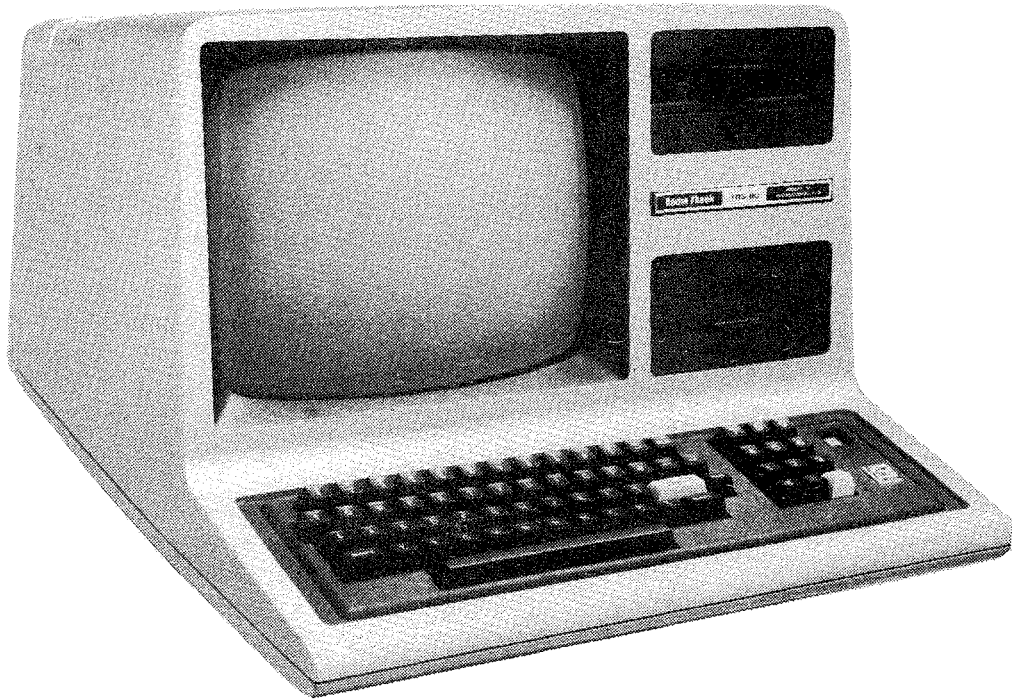
Daisy Wheel Typewriter/Printer

MICRO-80 has converted the new OLIVETTI ET-121 DAISY WHEEL typewriter to work with the TRS-80 and SYSTEM 80 or any other microcomputer with a Centronics parallel port (RS 232 serial interface available shortly). The ET-121 typewriter is renowned for its high quality, fast speed (17 c.p.s.), quietness and reliability. MICRO-80 is renowned for its knowledge of the TRS-80/SYSTEM 80 and its sensible pricing policy. Together, we have produced a dual-purpose machine: an attractive, modern, correcting typewriter which doubles as a correspondence quality Daisy-wheel printer when used with your micro-computer.

How good is it? - This part of our advertisement was typeset using an ET-121 driven by a TRS-80. Write and ask for full details.



1.4 MEGABYTES ON LINE + 48K RAM
for \$3800 incl. Sales Tax



MICRO-80's
MODEL 380 +

MICRO-80 has equipped the TRS-80 with two high reliability dual-head 80 track mini-floppy disk drives made by MPI, one of America's leading mini-disk drive manufacturers.

This turns the mild-mannered Model 3 into a powerhouse able to handle the most difficult business programs. The TRS-80 is one of the best-supported microcomputers in the world. MICRO-80 has been supporting the TRS-80 in Australia for 18 months and is one of Australia's leading dealers in MPI disk drives.

2.8 MEGABYTES FOR \$5300 incl. Sales Tax

If you need even more file space you can add MICRO-80's external dual-drive cabinet enclosing two more dual-head 80 track drives for an additional \$1500.

COMPUTER PRICES

MODEL 340

2 40 TRACK SINGLE HEAD DRIVES GIVING
350K FORMATTED STORAGE, 48K RAM

\$2990 INCL. SALES TAX

MODEL 340 +

2 40 TRACK DUAL-HEAD DRIVES GIVING
700K FORMATTED STORAGE, 48K RAM

\$3350 INCL. SALES TAX

MODEL 380

2 80 TRACK SINGLE HEAD DRIVES GIVING
700K FORMATTED STORAGE, 48K RAM

\$3350 INCL. SALES TAX

MODEL 380 +

2 80 TRACK DUAL-HEAD DRIVES GIVING
1.4 MEGABYTE FORMATTED STORAGE, 48K RAM

\$3800 INCL. SALES TAX

350K SYSTEM

MODEL 340, EPSON MX-80 PRINTER
NEWDOS 80 DISK OPERATING SYSTEM

\$4070 INCL. SALES TAX

700K SYSTEM (40 Track)

MODEL 340 +, EPSON MX-80 PRINTER
NEWDOS 80 DISK OPERATING SYSTEM

\$4429 INCL. SALES TAX

700K SYSTEM (80 Track)

MODEL 380, EPSON MX-80 PRINTER
NEWDOS 80 DISK OPERATING SYSTEM

\$4429 INCL. SALES TAX

1.4 MEGABYTE SYSTEM

MODEL 380 +, EPSON MX-80 PRINTER
NEWDOS 80 OPERATING SYSTEM

\$4880 INCL. SALES TAX

2.8 MEGABYTE SYSTEM

MODEL 380 +, DUAL EXTERNAL DRIVES,
MX-80 PRINTER, NEWDOS 80 OPERATING SYSTEM

\$6380 INCL. SALES TAX



EXATRON STRINGY FLOPPY — \$372.50 Incl. P&P

All Exatron Stringy Floppies sold by MICRO-80 include the special chained version of **HOUSEHOLD ACCOUNTS**, developed by Charlie Bartlett. When used on the ESF, this program is powerful enough to perform many of the accounting functions in a small business. Remember, the ESF comes complete with a comprehensive manual, a 2 way bus-extender cable, its own power supply and 10 wafers of mixed length. One wafer contains the Data Input/Output program and another the **HOUSEHOLD ACCOUNTS** program.

CAN'T MAKE UP YOUR MIND ABOUT THE ESF?

Then send in \$5.00 for a copy of the manual. We will refund your \$5.00 IN FULL when you purchase an ESF.



SOFTWARE BY AUSTRALIAN AUTHORS

All our software is suitable for either the SYSTEM 80 or the TRS-80

NEW SOFTWARE FROM MICRO-80 PRODUCTS

BUSINESS PROGRAMS

MICROMANAGEMENT STOCK RECORDING SYSTEM (L2/16K)

Cassette version \$29.95 + \$1.00 p&p
Stringy Floppy version \$33.95 + \$1.00 p&p

This system has been in use for 9 months in a number of small retail businesses in Adelaide. It is therefore thoroughly debugged and has been tailor made to suit the requirements of a small business. MICROMANAGEMENT SRC enables you to monitor the current stock level and reorder levels of 500 different stock items per tape or wafer. It includes the following features:—

- Add new items to inventory
- Delete discontinued items from inventory
- List complete file
- Search for any stock number
- Save data to cassette or wafer
- Load data from cassette or wafer
- Adjusts stock levels from sales results and receipt of goods
- List all items requiring reordering

We can thoroughly recommend this program for the small business with a L2/16K computer.

SCOTCH BRAND COMPUTING CASSETTES

Super-quality personal computing cassettes.

C-10 pack of 10 \$26.00 incl. p&p
C-30 pack of 10 \$28.00 incl. p&p

UTILITIES

S-KEY by Edwin Paay **\$15.95 plus 50c. p&p**

S-KEY is a complete keyboard driver routine for the TRS-80 and becomes part of the Level II basic interpreter. With S-KEY loaded the user will have many new features not available with the standard machine.

S-KEY features:

- * S-KEY provides an auto-repeat for all the keys on the keyboard. If any key is held down longer than about half a second, the key will repeat until it is released.
- * Graphic symbols can be typed direct from the keyboard, this includes all 64 graphic symbols available from the TRS-80/SYSTEM 80.
- * S-KEY allows text, BASIC commands and/or graphics to be defined to shifted keys. This makes programming much easier as whole commands and statements can be recalled by typing shift and a letter key.
- * Because S-KEY allows graphics to be typed directly from the keyboard, animation and fast graphics are easily implemented by typing the appropriate graphics symbols directly into PRINT statements.
- * S-KEY allows the user to LIST a program with PRINT statements containing graphics, properly. S-KEY does this by intercepting the LIST routine when necessary.
- * S-KEY allows the user to list an updated list of the shift key entries to the video display or line printer.
- * S-KEY can be disabled and enabled when required. This allows other routines which take control of the keyboard to run with S-KEY as well.

Each cassette has TRS-80, DISK and SYSTEM 80 versions and comes with comprehensive documentation.

BMON by Edwin Paay **\$19.95 plus 50c. p&p**
THE ULTIMATE HIGH MEMORY BASIC MONITOR
L2/16-48K

Our own personnel refuse to write BASIC without first loading this amazing machine language utility program into high memory! BMON Renumbers; Displays BASIC programs on the screen while they are still loading; tells you the memory locations of the program just loaded; lets you stop a load part-way through; merges two programs, with automatic renumbering of the second so as to prevent any clashes of line numbers; recovers your program even though you did type NEW: makes one program invisible while you work on a second (saves hours of cassette time!); lists all the variables used in the program; makes SYSTEM tapes; lets you Edit memory directly . . . the list goes on and on. Cassette comes with 16K, 32K and 48K versions, ready to load. Can anyone afford NOT to have BMON?

EDUCATIONAL

RPN CALCULATOR (L2/16K & 32K)
\$14.95 \$ 50c. p&p

Give your computer the power of a \$650 reverse polish notation calculator with 45 functions and selectable accuracy of 8 or 16 digits. The main stack and registers are continuously displayed whilst the menu is always instantly accessible without disturbing any calculations or register values. The cassette comes with both the 16K and 32K versions, the latter giving you the additional power of a programmable calculator. Comes with a very comprehensive 15 page manual, which includes instructions to load and modify the 32K programmable version to run in 16K. Whether for business or pleasure, this package will prove invaluable, and turn you '80 into a very powerful instrument.

GAMES

MICROPOLY (L2/16K) **\$8.95 + 60c p&p**

Now you can play Monopoly on your micro. The old favourite board game has moved into the electronic era. This computer version displays the board on the screen, obeys all the rules and, best of all, the banker does not make mistakes with your change!

CONCENTRATION (L2/16K) **\$8.95 + 60c p&p**

Another application of supergraphics. There are 28 "cards" displayed on the screen, face down. Players take it in turn to turn them over with the object of finding matching pairs. There are 40 different patterns which are chosen at random, so the game is full of endless variety. This is of particular value in helping young children to learn the art of concentrating and, at the same time, to introduce them to the computer.

METEOR AND TORPEDO ALLEY (L2/16K)
\$10.95 + 60c p&p

Those who frequent games arcades will recognize these two electronic games. In METEOR you must destroy the enemy space ships before they see you. In its most difficult mode, the odds are a thumping 238 to 1 against you being successful. In torpedo alley you must sink the enemy ships without hitting your own supply ship. Both games include sound effects and are remarkably accurate reproductions of the arcade games.

AUSTRALIAN SOFTWARE (Cont.)**GAMES****SHEEPDOG (L2/16K)****\$8.95 + 60c p&p**

Ever wondered how a sheepdog manages to drive all those awkward sheep into a pen? Well, here is your chance to find out just how difficult it is and have a lot of fun at the same time. You control the sheepdog, the computer controls the sheep! As if that isn't enough, look out for the dingoes lurking in the bush!

U BOAT**\$8.95 + 60c p&p**

Real time simulation at its best! Comes with working sonar-screen and periscope, a full rack of torpedoes, plenty of targets, working fuel and battery meters, helpful Mothership for high-seas provisioning and even has emergency radio for that terrible moment when the depth charges put your crew at risk. Requires Level II/16K.

SPACE INVADERS WITH SOUND**\$8.95 + 60c p&p**

Much improved version of this arcade favourite with redesigned laser and cannon blasts, high-speed cannon, 50 roving drone targets, 10 motherships and heaps of fun for all. Level II with 4K and 16K versions on this cassette.

GOLF (L2/16K)**\$8.95 + 60c p&p**

Pit your skills of mini-golf against the computer. Choose the level of difficulty, the number of holes and whether you want to play straight mini golf or crazy golf. Complete with hazards, water traps, bunkers and trees. Great fun for kids of all ages.

DOMINOES(L2/16K)**\$8.95 + 60c p&p**

Pit your skill at dominoes against the computer, which provides a tireless opponent. Another application of supergraphics from the stable of Charlie Bartlett. Dominoes are shown approximately life size in full detail (except for colour!). The monitor screen is a window which you can move from one end of the string of dominoes to the other. Best of all, you don't lose any pieces between games!

KID'S STUFF (formerly MMM-1)**\$8.95 + 60c p&p**

Three games on one cassette from that master of TRS-80 graphics, Charlie Bartlett. Includes INDY 500, an exciting road race that gets faster and faster the longer you play, SUBHUNT in which your warship blows up unfortunate little submarines all over the place, and KNEVEL (as in motorcycle, ramp and buses).

OTHER PROGRAMS**INFINITE BASIC BY RACET (32K/1 DISK)****\$49.95 + 50c. p&p**

Full matrix functions – 30 BASIC commands; 50 more STRING functions as BASIC commands.

GSF/L2/48K**\$24.95 + 50c. p&p**

18 machine language routines including RACET sorts.

BUSINESS ADDRESS AND INFORMATION SYSTEM (48K/DISK)**\$24.95 + 50c. p&p**

Allows you to store addresses and information about businesses, edit them and print them out.

HISPED (L216, 32 or 48K) \$29.95

This machine language program allows you to SAVE and LOAD programs and data to tape at speeds up to 2000 baud (4 times normal) using a standard cassette recorder. A switch must be installed to remove the XRX III loading board, if fitted.

LOWER CASE FOR YOUR TRS-80/SYSTEM 80**Kit only \$49.00 plus \$2.00 p&p**

Give your TRS-80 or SYSTEM 80 a lower case display with proper descenders and a block cursor (similar to the TRS-80 Model III). Also includes symbols for the four suits of cards. Includes full fitting instructions, all necessary components and a special machine language driver program to enable lower case in BASIC. The modification is similar to the Tandy model and does not work with Electric Pencil without further modifications.

These kits require disassembly of your computer and some soldering. They should only be installed by someone who has experience in soldering integrated circuits, using a low power, properly earthed soldering iron. If you do not have the necessary experience/equipment, we will install the modification for you for \$20 plus freight in both directions. Make sure you arrange the installation with us first, before despatching your computer, so that we can assure you of a rapid turn-around. We are also arranging to have installers in each State. See elsewhere in this issue for their names and addresses.

PRICES

Cat No.

HD-020 Lower case mod kit for TRS-80

\$49.00 plus \$2.00 p&p

HD-021 Lower case mod kit for SYSTEM-80

\$49.00 plus \$2.00 p&p**EPSON MX-80 PRINTER****ONLY *\$949 Inc. Cable for TRS-80 and p&p****(*Printer only – \$940 incl. p&p)**

The EPSON MX-80 printer is compact, quiet, has features unheard of only 2-3 years ago in a printer at any price and, above all, is ultra-reliable. All available print modes may be selected under software control. Features include:

- high quality 9x9 dot-matrix character formation
- 3 character densities
 - . 80 characters per line at 10 chars/inch
 - . 132 characters per line at 16.5 chars/inch
 - . 40 characters per line at 5 chars/inch
- 2 line spacings
 - . 6 lines per inch 8 lines per inch
- 80 characters per second print speed
- bi-directional printing
- logical seeking of shortest path for printing
- lower case with descenders
- TRS-80 graphics characters built in
- standard Centronics printer port

The bi-directional printing coupled with the logical seeking of the shortest print path (which means that the print head will commence printing the next line from the end which requires the least travel, thereby minimising unutilised time) gives this printer a much higher throughput rate than many other printers quoting print speeds of 120 c.p.s. or even higher.

GREEN SCREEN SIMULATOR**\$9.50 incl. p&p**

The GREEN SCREEN SIMULATOR is made from a deep green perspex, cut to fit your monitor. It improves contrast and is much more restful to the eyes than the normal grey and white image.

All editorial staff of MICRO-80 are now using GREEN SCREEN SIMULATORS on their own monitors.

Please make sure to specify whether you have an old (squarish) or new (rounded) style monitor when ordering. Not available for Dick Smith monitors.

The syntax for calling one of the functions is $X = \text{USR function-ID (Y)}$ where X and Y (which must be variable names conforming to BASIC rules, and not expressions or numeric values) have previously been defined in the program as double precision variables.

Thus, $X = \text{USR SIN(Y)}$ returns X as $\sin(Y)$, and similarly for COS , TAN , ATN , EXP , LOG and SQR : additionally, $X = \text{USR D(Y)}$ returns X as the degree equivalent of Y radians, $X = \text{USR R(Y)}$ returns X as the radian equivalent of Y degrees, and $X = \text{USR P(Y)}$ returns X as the double precision value of π . Note in the last case that the dummy variable Y is still required, even though it is not operated upon, so that $X = \text{USR P(X)}$ would be just as valid as a call. Note also, from the example just given, that spaces are not essential in the calls; they have only been included for clarity in the earlier examples.

These functions are based on the summation of convergent infinite series until the next term is of the order of 2^{-129} ; dependent on the value of the independent variable, this can mean quite a few summations, so that the functions will sometimes take a noticeable time to return a result. A call that would yield an imaginary result (such as $\log(-1)$) or an infinite result (such as $\tan 90$ deg.) will return SN ERROR or $/\emptyset \text{ ERROR}$ as appropriate. Excess values will return the normal OV ERROR : there is also some loss of accuracy, particularly of the LOG and SQR functions when the independent variable is very large or very small. SIN , COS , TAN and ATN functions refer to angles in radians.

The calling routine is an adaptation of the HMT Corpn, expanded USR function program, and there are still 60 bytes available in the address index table for further USR functions; since one function uses $(\text{length of function-ID})+3$ bytes, this is sufficient for 15 functions with single character names (or BASIC reserved words, which store as single characters), 12 with two character names, and so on. The rules for adding your own functions are:-

1. The calling syntax is $A = \text{USR function-ID(B,C,D,...)}$ where A, B, C, D , etc. are properly named and defined BASIC variables; there must be at least one such argument (variable name) following the function-ID in brackets, even if it is a dummy; otherwise you may have as many as you like or need.
2. Somewhere in the calling program, prior to the first call of the function, there must be an entry point definition in the form $\text{DEF function-ID} = \text{nnnnn}$ where nnnnn is the decimal address, or an expression reducing to the decimal address, of the entry point of the function.
3. The DEF portion of DUPLEX records this address against the function-ID in its address index table.
4. The USR portion will stuff an address (2 bytes) into a store, for each of the variables in the string of arguments appearing in brackets after the function-ID; it then sets the IX register to point to the first address in the store and jumps to the entry point of your routine. The addresses stored in regard to the variables are those that would be returned by VARPTR for that variable.
5. If you want the variable on the left of the $=$ sign in the call to contain a value on return, your routine must load the value into the ACCumulator at 411DH to 4124H , and set the NTF in 40AFH before returning; otherwise you can transfer values by loading them (in form appropriate to the type of variable) to the address provided by the USR portion of the package, and stuffed into the store to which the IX register is pointing.

The foregoing set of rules may sound horrific, but it's not really. If you have a simple routine to, say, white out the screen, which you wish to use, you:

- (a) give it a name - say, BLOB
- (b) put a statement early in your program that says
 $\text{DEF BLOB} = \text{xxxxx}$
 where xxxxx is the decimal address of the entry point of BLOB .
- (c) call the routine with
 $\text{DUMMY} = \text{USR BLOB(DUMMY)}$
 whenever you need it.

Some people may want the DEF and USR portions of DUPLEX without the DP mathematical functions. If so, delete lines 2000 to 5530 , and 5800 to 6010 , (all inclusive) in the assembler program and re-assemble. You will then have the DEF and USR facilities, and a 100 byte address index table.

As regards the actual math portion of the program, there is not a great deal that can be easily explained. It relies heavily on a cluster of subroutines (which really form one enormous multiple entry point subroutine). Apart from that, there are two iterative calculation loops, one for EXP , SIN and COS , and one for ATN and LOG ; TAN and SQR are derived functions ($\text{TAN} = \text{SIN}/\text{COS}$ and $\text{SQR} = \text{EXP}(\frac{1}{2}\text{LOG})$). A value of π had to be stored for ATN , so that P , D and R functions are by-products.

```

10 REM: MEASUREMENTS-1.
20 REM PROGRAMMER-G. SKORYK
30 CU$="CUBE ":RE$="RECTANGLE ":RH$="RHOMBUS ":TR$="TRIANGLE ":
  PA$="PARALLELOGRAM ":TP$="TRAPESIUM ":CY$="CYLINDER ":
  PY$="PYRAMID ":CO$="CONE ":SP$="SPHERE "
40 BA$="FACE ":FA$="BASE ":PM$="PRISM ":BP$="FACE PRISM ":
  AR$="AREA ":SU$="SURFACE ":SA$="SURFACE AREA "
50 EN$="ENTER ":LE$="LENGTH ":SI$="SIDES ":HE$="HEIGHT ":
  RA$="RADIUS ":DI$="DIAGONALS ":V$="VOLUME ":T$="DEEP "
60 O$="OF ":AN$="AND ":F$="FOR ":W$="WHAT ":H$="HAS ":
  :A$="A ":WI$="WITH ":WH$="WHOLE ":I$="IS "
70 NR=0
80 NW=0
90 GOSUB 190
100 A=0: A1=0: V=0: V1=0: S=0: S1=0
110 ON K GOSUB 300,290
120 ON T GOSUB 450,490,530,590,620,650, 690,710,730
130 ON T GOSUB 760,770,780,800,810,820,850,860,880,900
140 INPUT "PRESS 'NEWLINE' "; Z:CLS:PRINT@ 0,"YOUR PROGRESS SCORE I
S";NR;"OUT OF";NR+NW
150 INPUT "TO CONTINUE PRESS '1' ; TO QUIT PRESS '0' ";Q
160 IF Q>1 OR Q<0 THEN 150
170 IF Q=1 THEN 90
180 IF Q=0 THEN CLS:END
190 CLS:PRINT "M E A S U R E M E N T S - 1."
200 PRINT "=====
210 PRINT "1-CUBE 2- " ;RE$;BP$
220 PRINT "3-RHOMBUS FACE PRISM 4- " ;TR$;BP$
230 PRINT "5-PARALLELOGRAM FACE PRISM 6- " ;TP$;BP$
240 PRINT "7-CYLINDER 8- " ;PY$
250 PRINT "9-CONE 10- " ;SP$
260 INPUT "WHICH ONE"; T: IF T<1 OR T>10 THEN 260
270 INPUT "(1) CHECK HOMEWORK OR (2) PRACTICE";K: IF K<1 OR K>2 T
HEN 270
280 RETURN
290 RANDOM: ON T GOSUB 310,310,330,310,310,310,310,310,310,310:RE
TURN
300 ON T GOSUB 350,360,370,380,390,400,410,420,430,440:RETURN
310 RX=RND(8):HX=RND(8):EX=RND(8):BX=RND(8):LX=RND(8)
320 R=RX+1:H=HX+1:E=EX+1:B=BX+1:L=10*(LX+1):RETURN
330 RX=RND(5):HX=RND(5):LX=RND(8):IF RX=HX THEN 330
340 R=RX+5:H=HX+5:L=10*(LX+1):RETURN
350 PRINT F$ CU$ EN$ LE$ O$ SI$:INPUT R:RETURN
360 PRINT F$ PM$ EN$ SI$ O$ BA$ RE$:INPUT R,H:GOSUB 960:RETURN
370 PRINT F$ PM$ EN$ DI$ O$ RH$ BA$:INPUT R,H:GOSUB 960:RETURN
380 PRINT F$ PM$ EN$ LE$ AN$ HE$ O$ TR$ BA$:INPUT R,H:GOSUB 960:
RETURN
390 PRINT F$ PM$ EN$ LE$ AN$ HE$ O$ PA$ BA$:INPUT R,H:GOSUB 960:
RETURN
400 PRINT F$ PM$ EN$ LE$ O$;"PARALLLEL ";SI$ "
" AN$ HE$ O$ TP$ BA$ :INPUT R,E,H:GOSUB 960:RETURN
410 PRINT F$ CY$ EN$ RA$ O$ FA$:INPUT R:GOSUB 960:RETURN
420 PRINT F$ PY$ EN$ AR$ O$ FA$:INPUT R:GOSUB 960:RETURN
430 PRINT F$ CO$ EN$ RA$ O$ FA$:INPUT R:GOSUB 960:RETURN
440 PRINT F$ SP$ EN$ RA$:INPUT R:RETURN

```

```

450 A1=R*R: S1=6*R*R: V1=R*R*R:RETURN
460 PRINT "WRONG !-AREA OF FACE=";R;"*";R;"=";A1:RETURN
470 PRINT "WRONG !-VOLUME=";R;"*";R;"*";R;"=";V1:RETURN
480 PRINT "WRONG !-WHOLE SURFACE AREA=6*";R;"*";R;"=";S1:RETURN
490 A1=R*H: S1=2*R*H+2*R*L+2*H*L: V1=R*H*L:RETURN
500 PRINT "WRONG !-" AR$ O$ BA$="";R;"*";H;"=";A1:RETURN
510 PRINT "WRONG !-VOLUME=";R;"*";H;"*";L;"=";V1:RETURN
520 PRINT "WRONG !-" WH$ SA$="";PRINT "2*";R;"*";H;"+"2*";R;"*";L;"
+2*";H;"*";L;"=";S1:RETURN
530 A1=R*H/2: V1=R*H/2*L: SI=((.5*R) [2+(.5*H) [2] [.5: S1=2*A1+4*SI*L
:RETURN
540 PRINT "WRONG !-" AR$ O$ BA$="";R;"*";H;"/2=";A1:RETURN
550 PRINT "WRONG !-VOLUME=";A1;"*";L;"=";V1:RETURN
560 PRINT "WRONG !-" WH$ SA$="2*";A1;"+"4*";SI;"*";L;"=";S1:RETURN
570 PRINT "WRONG !-" AR$ O$ BA$="";R;"*";H;"/2=";A1:RETURN
580 PRINT "WRONG !-VOLUME=";A1;"*";L;"=";V1:RETURN
590 A1=R*H: V1=R*H*L:RETURN
600 PRINT "WRONG !-" AR$ O$ BA$="";R;"*";H;"=";A1:RETURN
610 PRINT "WRONG !-VOLUME=";A1;"*";L;"=";V1:RETURN
620 A1=H/2*(R+E): V1=H/2*(R+E)*L:RETURN
630 PRINT "WRONG !-" AR$ O$ BA$="";H;"/2*(";R;"+";E;")=";A1:RETUR
N
640 PRINT "WRONG !-VOLUME=";A1;"*";L;"=";V1:RETURN
650 A1=22/7*R*R: S1=2*22/7*R*L+2*22/7*R*R: V1=22/7*R*R*L:RETURN
660 PRINT "WRONG !-" AR$ O$ FA$="22/7 *";R;"*";R;"=";A1:RETURN
670 PRINT "WRONG !-VOLUME=";A1;"*";L;"=";V1:RETURN
680 PRINT "WRONG !-" WH$ SA$="2*";A1;"+" 2* 22/7*";R;"*";L;"=";S1
:RETURN
690 V1=B*L/3:RETURN
700 PRINT "WRONG !-VOLUME=";B;"*";L;"/3=";V1:RETURN
710 V1=22/7*R*R*L/3:RETURN
720 PRINT "WRONG !-VOLUME=22/7*";R;"*";R;"*";L;"/3=";V1:RETURN
730 S1=4*22/7*R*R: V1=4*22/7*R*R*R/3:RETURN
740 PRINT "WRONG !-VOLUME=4/3* 22/7*";R;"*";R;"*";R;"=";V1:RETURN
750 PRINT "WRONG !-" WH$ SA$="4* 22/7*";R;"*";R;"=";S1:RETURN
760 GOSUB 1070:PRINT A$ CU$ H$ SI$;R;" LONG " :
GOSUB 910:GOSUB 970:GOSUB 1020:RETURN
770 GOSUB 1240:PRINT A$ PM$ H$ RE$ BA$ WI$ SI$ ;R; AN$ ;H:PRINT
"IT " I$ ;L; T$:GOSUB 910:GOSUB 970:GOSUB 1020:RETURN
780 GOSUB 1390:PRINT A$ PM$ H$ RH$ BA$ WI$ DI$ ;R; AN$ ;H;"
(SIDES ARE ";SI;"LONG)"
790 PRINT "IT IS";L;"DEEP ":GOSUB 910:GOSUB 970:GOSUB 1020:RETURN
800 GOSUB 1710:PRINT A$ PM$ H$ TR$ BA$ ;R; "BY ";H; "HIGH "AN$ I
$ ;L; T$:GOSUB 910:GOSUB 970:RETURN
810 GOSUB 1560:PRINT A$ PM$ H$ PA$ BA$ ;R; "BY ";H; "HIGH ":PRIN
T "IT " I$ ;L; T$:GOSUB 910:GOSUB 970:RETURN
820 GOSUB 1840:PRINT A$ PM$ H$ TP$ BA$ WI$ "PARALLEL " SI$ ;R; A
N$ ;E
830 PRINT "THEY ARE ";H;" APART.THIS PRISM IS ";L;"DEEP"
840 GOSUB 910:GOSUB 970:RETURN
850 GOSUB 2000:PRINT A$ CY$ H$ FA$ RA$ ;R; AN$ I$ ;L; T$:GOSUB 9
10:GOSUB 970:GOSUB 1020:RETURN
860 GOSUB 2180:PRINT A$ PY$ H$ FA$ AR$ ;R; AN$ I$ ;L; T$
870 GOSUB 970:RETURN

```

```

880 GOSUB 2070:PRINT A$ CO$ H$ FA$ RA$ ;R; AN$ I$ ;L; T$
890 GOSUB 970:RETURN
900 GOSUB 2130:PRINT A$ SP$ H$ RA$ ;R:GOSUB 970:GOSUB 1020:
  RETURN
910 INPUT"WHAT IS AREA OF FACE"; A
920 IF ABS(A1-A)<=A1/100 THEN 950
930 ON T GOSUB 460,500,540,570,600,630,660,700,720,740
940 NW=NW+1:RETURN
950 PRINT"RIGHT !":NR=NR+1:RETURN
960 INPUT"HOW DEEP";L:RETURN
970 INPUT"WHAT IS THE VOLUME";V
980 IF ABS(V1-V)<=V1/1000 THEN 1010
990 ON T GOSUB 470,510,550,580,610,640,670,700,720,
  740
1000 NW=NW+1:RETURN
1010 PRINT"RIGHT !":NR=NR+1:RETURN
1020 INPUT"WHAT IS WHOLE SURFACE AREA";S
1030 IF ABS(S1-S)<=S1/1000 THEN 1060
1040 ON T GOSUB 480,520,560,580,610,640,680,700,720,740
1050 NW=NW+1:RETURN
1060 PRINT"RIGHT !":NR=NR+1:RETURN
1070 M1=2
1080 IF R<10 THEN 1100
1090 H1=9:R1=9:L=45:GOTO 1110
1100 H1=R:R1=R:L=5*R
1110 X1=109
1120 X2=X1-2*M1*R1
1130 X3=X2
1140 X4=X1
1150 Y1=40
1160 Y2=Y1-R1*M1
1170 SL=0
1180 SP=SL
1190 GOSUB 2380
1200 PRINT@ 887,R
1210 PRINT@(875-.5*X1+.5*X2),R
1220 PRINT@ 948,R
1230 RETURN
1240 GOSUB 2330
1250 H1=H:R1=R
1260 X1=109
1270 X2=X1-2*M1*R1
1280 X3=X2
1290 X4=X1
1300 Y1=40
1310 Y2=Y1-H1*M1
1320 SL=0
1330 SP=SL
1340 GOSUB 2380
1350 PRINT@ 887,H
1360 PRINT@(875-.5*X1+.5*X2),L
1370 PRINT@ 948,R
1380 RETURN
1390 GOSUB 2330
1400 AN=ATN(D2/D1)
1410 H1=D1*SIN(AN)

```

```

1420 R1=((.5*R)[2+(.5*H)[2].5
1430 X1=109
1440 X2=X1-2*M1*R1
1450 X3=X1-2*M1*D1*ICOS(AN)
1460 X4=X3+2*M1*R1
1470 Y1=40
1480 Y2=Y1-H1*M1
1490 SL=(X2-X3)/(Y1-Y2)
1500 SP=SL
1510 GOSUB 2380
1520 FOR X=0 TO (X1-X3):Y=(Y1-Y2)/(X1-X3)*X:SET(X1-X,Y1-Y):NEXT
  X
1530 FOR X=0 TO (X4-X2):Y=(Y1-Y2)/(X4-X2)*X:SET(X2+X,Y1-Y):NEXT
  X
1540 PRINT@930,"DIAGONALS ARE";R;"AND";H
1550 RETURN
1560 GOSUB 2330
1570 H1=H:R1=R
1580 X1=109
1590 X2=X1-2*M1*R1
1600 Y1=40
1610 Y2=Y1-H1*M1
1620 SL=1
1630 SP=SL
1640 X3=X2-(Y1-Y2)
1650 X4=X3+(X1-X2)
1660 GOSUB 2380
1670 PRINT@ 887,H
1680 PRINT@(875-.5*X1+.5*X2),L
1690 PRINT@ 948,R
1700 RETURN
1710 GOSUB 2330
1720 H1=H:R1=R
1730 X1=109
1740 X2=X1-2*M1*R1
1750 Y1=40:Y2=40-M1*H1
1760 S3=RND(2)
1770 IF S3=1 THEN 1790
1780 X3=X2-Y1+Y2:SL=1:SP=(X1-X3)/(Y1-Y2):X4=X3:GOTO 1800
1790 X3=X2+Y1-Y2:X4=X3:SL=-1:SP=(X1-X3)/(Y1-Y2)
1800 GOSUB 2380
1810 PRINT@(875-.5*X1+.5*X2),L
1820 PRINT@ 887,H:PRINT@ 948,R
1830 RETURN
1840 IF R>E THEN 1860
1850 D1=E:D2=R:GOTO 1870
1860 D1=R:D2=E
1870 IFH>D1 THEN M1=20/H ELSE M1=20/D1
1880 X1=103:X2=X1-2*M1*D1
1890 Y1=40:Y2=Y1-M1*H
1900 S3=RND(2)
1910 IF S3=1 THEN 1930
1920 X3=X2-Y1+Y2:SL=1:X4=X3+2*M1*D2:SP=(X1-X4)/(Y1-Y2):GOTO 1940
1930 X3=X2+Y1-Y2:SL=-1:X4=X3+2*M1*D2:SP=(X1-X4)/(Y1-Y2)
1940 GOSUB 2380
1950 ON ERROR GOTO 1970

```

```

1960 PRINT@ (878-63*(INT((Y1-Y4)/3)+3)-.5*(X1-X6+4)),D2
1970 PRINT@ (873-.5*X1+.5*X2),L
1980 PRINT@885,H:PRINT@ 946,D1
1990 RETURN
2000 INPUT"PRESS 'NEW LINE'";Z:CLS
2010 Y1=34:GOSUB 2310:GOSUB 2320
2020 Y1=20:GOSUB 2320
2030 X=68:FOR Y=20 TO 34:SET(X,Y):NEXT Y
2040 X=116:FOR Y=20 TO 34:SET(X,Y):NEXT Y
2050 X=92:FOR Y=34 TO 40 STEP 2:SET(X,Y):NEXT Y
2060 PRINT@ 505,L:PRINT@ 941,R:RETURN
2070 INPUT"PRESS 'NEW LINE'";Z:CLS
2080 Y1=34:GOSUB 2310:GOSUB 2320
2090 FOR X=0 TO 24:Y=-X:SET(X+68,Y+34):NEXT X
2100 FOR X=0 TO 24:Y=X:SET(X+92,Y+10):NEXT X
2110 X=92:FOR Y=34 TO 40 STEP 2:SET(X,Y):NEXT Y
2120 PRINT@365,L:PRINT@941,R:RETURN
2130 INPUT"PRESS 'NEW LINE'";Z:CLS
2140 FOR X=-24 TO 24:Y=.5*(24[2-X[2].5:SET(X+92,Y+28):NEXT X
2150 FOR X=-24 TO 24:Y=-.5*(24[2-X[2].5:SET(X+92,Y+28):NEXT X
2160 X=92:FOR Y=28 TO 40 STEP 2:SET(X,Y):NEXT Y
2170 PRINT@ 941,R:RETURN
2180 INPUT"PRESS 'NEW LINE'";Z:CLS
2190 Y=40:FOR X=80 TO 104:SET(X,Y):NEXT X
2200 FOR X=0 TO 12:Y=-.5*X:SET(X+104,Y+40):NEXT X
2210 FOR X=12 TO 0 STEP -1:Y=.5*X:SET(X+104,Y+28):NEXT X
2220 Y=28:FOR X=104 TO 80 STEP -1:SET(X,Y):NEXT X
2230 FOR X=12 TO 0 STEP -1:Y=-.5*X:SET(X+68,Y+34):NEXT X
2240 FOR X=0 TO 12:Y=.5*X:SET(X+68,Y+34):NEXT X
2250 FOR X=0 TO 24:Y=-X:SET(X+68,Y+34):NEXT X
2260 FOR X=0 TO 24:Y=X:SET(X+92,Y+10):NEXT X
2270 FOR X=0 TO 12:Y=-18/12*X:SET(X+80,Y+28):NEXT X
2280 FOR X=0 TO 12:Y=18/12*X:SET(X+92,Y+10):NEXT X
2290 PRINT@ 896,"AREA OF BASE=";B;" HEIGHT=";L
2300 RETURN
2310 FOR X=-24 TO 24:Y=.25*(24[2-X[2].5:SET(X+92,Y+Y1):NEXT X:
RETURN
2320 FOR X=-24 TO 24:Y=-.25*(24[2-X[2].5:SET(X+92,Y+Y1):NEXT X:
RETURN
2330 IF R>H THEN 2350
2340 D1=H:D2=R:GOTO 2360
2350 D1=R:D2=H
2360 M1=20/D1
2370 RETURN
2380 INPUT"PRESS 'NEW LINE'";Z:CLS
2390 Y=Y1:FOR X=X1 TO X2 STEP -1:SET(X,Y):NEXT X
2400 FOR Y=Y1 TO Y2 STEP -1:X=X2+SL*(Y-Y1):SET(X,Y):NEXT Y
2410 IF X3=X4 THEN 2430
2420 Y=Y2:FOR X=X3 TO X4:SET(X,Y):NEXT X
2430 FOR Y=Y2 TO Y1:X=X4+SP*(Y-Y2):SET(X,Y):NEXT Y
2440 IF L>70 THEN LX=35 ELSE LX=L/2
2450 X5=X2-LX:X6=X3-LX:X7=X4-LX
2460 Y3=Y1-LX/5:Y4=Y3-Y1+Y2
2470 FOR Y=Y3 TO Y4 STEP -1:X=X5+SL*(Y-Y3):SET(X,Y):NEXT Y
2480 IF X6=X7 THEN 2500

```

```

2490 Y=Y4:FOR X=X6 TO X7:SET(X,Y):NEXT X
2500 FOR X=X2 TO X5 STEP-5:Y=Y1+(X-X2)/5:SET(X,Y):NEXT X
2510 FOR X=X3 TO X6 STEP-5:Y=Y2+(X-X3)/5:SET(X,Y):NEXT X
2520 IF X6=X7 THEN 2540
2530 FOR X=X4 TO X7 STEP-5:Y=Y2+(X-X4)/5:SET(X,Y):NEXT X
2540 RETURN
2550 FOR Z=1 TO 100 :NEXT Z:RETURN

```

```

10 REM *****
SUPER-HANGMAN : (C) AUG. 1981 MARTIN DOWNEY
*****
20 CLS:OUT254,255
30 FORI=0TO15:FORJ=0TO9:PRINT@ (I*64+2+J*6),"SUPER";:NEXTJ,I
40 PRINT@468,STRING$(24," ");:PRINT@660,STRING$(24," ");
50 PRINT@532," H A N G M A N ";:CLEAR2500:ZZ=1
60 PRINT@596," (SIGHT & SOUND) ";
70 PRINT@846," (C) AUG. 1981 : MARTIN DOWNEY ";
80 DEFINTA-Y:DIMG$(40),D$(19),D(27)
90 RANDOM:RESTORE:GOSUB350:CLS:GOSUB770
100 L=0:II=4:GOSUB780:GOSUB2100:IFZZ=0THEN290
110 LZ=4:JZ=13:KZ=10:T$=T$(1):IZ=1
120 PRINT@20,"WELCOME TO SUPER-HANGMAN";:
PRINT@76,"PRESS <I> FOR INSTRUCTIONS OR <B> TO BEGIN";
130 GOSUB1760:I$=INKEY$:IZ=IZ+1:IFI$="B"THEN290
ELSEIFIZ<B1THEN130
140 GOSUB1830:GOSUB1840:KK=1:LL=30:GOSUB2070:KK=4:GOSUB2070
150 PRINT@12,"I AM A MEAN, UGLY, SELFISH, ROTTEN BANDIT."
160 JZ=17:LZ=50:KZ=15:T$=T$(4)
170 FORIZ=1TO31:GOSUB1760:NEXTIZ:PRINT@87,"(NOBODY'S PERFECT)";
180 FORL=1TO900:NEXTL
190 PRINT@12,"BUT I AM GOING TO BE GENEROUS FOR A CHANGE"
200 KK=7:GOSUB2070:KK=1:GOSUB2070:KK=4:GOSUB2070
210 PRINT@70,"AND GIVE YOU A CHANCE TO SAVE YOURSELF FROM HANGIN
G.":FORL=1TO60:H=61-L:GOSUB1800
220 NEXTL:PRINT@6,"I WILL THINK OF A WORD AND YOU MUST GUESS WHA
T IT IS."
230 PRINT@70,"YOU MAY GUESS ONE LETTER AT A TIME OR THE WHOLE WO
RD. ";
240 FORL=1TO110:GOSUB1700:NEXTL:GOSUB2100:II=4:GOSUB780
250 PRINT@10,"IF YOU MAKE SIX WRONG GUESSES YOU WILL LOSE."
260 GOSUB1830:GOSUB1840
270 FORL=1TO750:NEXTL:PRINT@87,"G O O D L U C K";
280 FORH=40TO10STEP-10:L=70:GOSUB1800:NEXTH:
FORL=1TO500:NEXTL
290 GOSUB2100:PRINT@21,"JUNIOR OR SENIOR LEVEL?"
300 PRINT@84,"PRESS <J> OR <S> TO START";
310 L=1
320 LV$=INKEY$:IFLV$="J"ORLV$="S"THEN340
330 GOSUB1700:L=L+1:IFL>300THEN100ELSE320
340 GOSUB2100:GOSUB2120:GOSUB790:GOTO830
350 FORJ=1TO7
360 FORI=1TO9:READX:A$(J)=A$(J)+CHR$(X+128):NEXTI
370 FORI=1TO7:READX:B$(J)=B$(J)+CHR$(X+128):NEXTI
380 FORI=1TO6:READX:C$(J)=C$(J)+CHR$(X+128):NEXTI

```

```

390 NEXTJ
400 DATA 48,60,60,63,63,63,60,60,52, 8,62,2,0,11,63,5,
    3,36,0,40,1,0
410 DATA 0,60,52,62,63,63,61,56,60, 0,2,60,1,2,47,5,
    2,36,1,32,6,0
420 DATA 0,52,48,62,61,63,61,48,56, 0,32,7,60,1,11,16,
    2,38,0,1,24,1
430 DATA 8,48,48,62,61,63,52,48,24, 0,26,2,60,22,10,16,
    9,18,1,3,24,1
440 DATA 0,52,48,62,63,62,61,48,56, 0,32,7,2,60,11,16,
    2,36,2,0,25,1
450 DATA 0,60,52,62,63,63,52,60,20, 0,47,7,0,41,22,0,
    2,36,0,33,6,0
460 DATA 56,60,60,63,63,63,60,60,52, 0,47,31,1,2,42,28,
    0,41,0,32,6,1
470 FORJ=OT03
480 FORI=1TO9:READX:D$(J)=D$(J)+CHR$(X+128):NEXTI,J
490 FORJ=4TO11:K=13:IFJ=5THENK=16
500 FORI=1TOK:READX:D$(J)=D$(J)+CHR$(X+128):NEXTI,J
510 FORI=OT027:READX:D(I)=X+128:NEXTI
520 DATA 0,56,63,59,60,54,63,60,16, 2,19,31,27,27,27,59,0,0
530 DATA 0,10,63,31,15,47,63,0,0, 48,48,63,4,0,8,63,48,48
540 DATA 32,48,60,14,63,47,60,63,47,31,44,52,48
550 DATA 2,15,60,16,29,25,25,59,0,0,0,3,11,28,16
560 DATA 0,0,56,28,63,47,60,63,47,13,60,16,0
570 DATA 0,0,2,19,31,27,25,27,63,3,3,0,0
580 DATA 48,56,12,15,63,47,60,63,47,15,15,44,52
590 DATA 0,3,13,16,29,25,25,59,0,0,14,1
600 DATA 56,28,15,15,63,47,60,63,47,15,47,52,0
610 DATA 2,3,20,0,29,25,25,59,0,56,7,0
620 DATA 60,31,15,0,2,9,0, 32,63,15,0,11,37,0
630 DATA 0,32,63,12,44,30,5, 0,32,63,3,61,14,5
640 T$(0)="KKKZKKKZKKHZGFZKKKZKKHZFFGZIKZKKKZKKKZKKHZGFZHFZDZZ
EFGHZFFHZKMKZMMZMMZLZKZMMZMMZMMKZIZHZZKZJZHZZOZNMZZZZZDDHZZD
IZDDFZZZZZ"
650 T$(1)="ABACDCEFEHZZJIHGFEDCBZDZBCBDEDFGFHZZABA@ABCDEZZZZDDDG
GGEEEFZDDDGEEFEFZDDDCDEFGHZZZZZZZ"
660 T$(2)="LLJHJLMZGHZZMMKIKMLZGHZLLJHJLMZGHZZMMKIKMLZHEZZGZGZ
ZGGIGEZHZZHZZHHJHEZIZZZI IKKHJJ IGHFEZA@ZZZZZZZZ"
670 T$(3)="IHIFZFFZFIHZZDZEDZIFFFFFFIHDZEDZZIFZFFFZIZDDEDZIHZFFF
IDZEDZIHZF5ZZZE5F5ZZZZZZZZ"
680 T$(4)="FFEDFFGZFFEDFZGZFFEDCDEFGIHFZFFZ"
690 WW$="WOW-WOW-WOW " : WW$=WW$+WW$+WW$+WW$+WW$
700 AR$=STRING$(32,32)+"<---<<<" +STRING$(32,32)
710 FORI=OT05:FORJ=1TO4:READX:S$(I)=S$(I)+CHR$(128+X):NEXTJ,I
720 FORI=OT09:READX,Y:Z=X+256*Y:IFZ>32768THENZ=Z-65536
730 M%(I)=Z:NEXTI:RETURN
740 DATA 24,1,2,36, 2,36,24,1, 32,6,9,16, 2,36,24,1
750 DATA 0,26,37,0, 0,37,26,0
760 DATA0,205,127,10,62,6,211,255,69,16,254,61,254,5,40,246,37,2
00,24,240
770 FORI=OT03:PRINT@405+I*64," " ;D$(I);" " ;:NEXTI
:RETURN
780 PRINT@218,A$(II);:PRINT@283,B$(II);:PRINT@348,C$(II);
:RETURN

```

```

790 FORX=30TO98:SET(X,39):SET(X,46):NEXTX
800 FORY=40TO45:SET(30,Y):SET(31,Y):SET(97,Y):SET(98,Y):NEXTY
810 PRINT@926-LEN(W$)," " ;:FORK=1TO2*LEN(W$):PRINTG$(K);:NEXTK
820 RETURN
830 I$=INKEY$
840 PRINT@15,"PRESS A LETTER KEY FOR YOUR GUESS"
850 PRINT@79,"PRESS THE @ KEY TO GUESS THE WORD"
860 PRINT@298,STRING$(21,32);:I$=INKEY$:GOSUB1700
:FORI=1TO10:NEXTI
870 F2=F2+1:IFF2=10THENF2=0:GOSUB2100:FORI=1TO19:NEXTI
880 IFI$=""THENIFF2=0THEN840ELSE860
890 X=ASC(I$):IF(X<64)OR(X>90)THEN860
900 F=-1:GOSUB1700:IFX=64THEN1010
910 PRINT@298,I$;:FORII=1TORND(5)STEP-1:GOSUB780
920 FORJ=1TO20:NEXTJ,II:KK=7:LL=20:GOSUB2070:GOSUB1830
930 FORI=1TOLEN(G$):IFMID$(G$,I,1)=I$THEN1000ELSENEXTI
940 FORI=1TO2*LEN(W$):IFG$(I)=I$THEN1200ELSENEXTI
950 G=0:FORI=1TOLEN(W$):IFMID$(W$,I,1)=I$THENG$(2*I)=I$:G=1
960 NEXTI:IFG=0THEN1140
970 GOSUB1960:GOSUB810:FORK=1TO80:NEXTK:GOSUB2050:GOSUB1840
980 G=0:FORI=1TO2*LEN(W$):IFG$(I)=CHR$(95)G=1
990 NEXTI:IFG=0THEN1300ELSE860
1000 M$="YOU'VE ALREADY TRIED THAT LETTER!":GOTO1210
1010 GOSUB2100
:PRINT@78,"TYPE IN YOUR WORD GUESS THEN PRESS <ENTER>";
1020 IK$="" :PRINT@298,CHR$(95);
1030 I$=INKEY$:GOSUB1700:IF(I$="" )OR((IK$="" )AND(I$=CHR$(8)))
THEN1030
1040 IFASC(I$)=8THENIK$=LEFT$(IK$,LEN(IK$)-1):I$=""
1050 IFI$=CHR$(13)THEN1080
1060 IK$=IK$+I$:PRINT@298,IK$;CHR$(95);" " ;
1070 IFLEN(IK$)<20THEN1030
1080 F=-1:GOSUB1700:KK=7:LL=20:GOSUB2070:GOSUB1830:I=1
1090 IFMID$(IK$,I,1)<>" "THEN1110
1100 IK$=LEFT$(IK$,I-1)+RIGHT$(IK$,LEN(IK$)-I):I=I-1
1110 I=I+1:IFI<=LEN(IK$)THEN1090
1120 WG$=IK$+STRING$(20-LEN(IK$),32):PRINT@298,WG$;
1130 IFIK$=W$THEN1290ELSEG1$=G1$+IK$+" " :GOTO1150
1140 G$=G$+I$+" "
1150 XG=XG+1:IFXG=6THEN1690
1160 GOSUB1860:FORI=1TO50:OUT255,6:FORJ=1TO(S6N(I-20)+1)*1.5:
NEXT:OUT255,5:NEXT:ONXG6GOSUB1530,1540,1560,1570,1600
1170 PRINT@640,G$;:PRINT@704,G1$;:IFXG=5THEN1190
1180 GOSUB2050:FORJ=1TORND(40):NEXTJ:GOSUB1840:GOTO860
1190 M$="LAST CHANCE COMING UP !!":GOTO1210
1200 M$="YOU'VE ALREADY GOT THAT LETTER!"
1210 KK=4:LL=30:GOSUB2070:GOSUB2100
1220 FORK=1TO5:PRINT@81,M$;:L=K*10:H=K*10:GOSUB1800
1230 FORL=1TOK*30:NEXTL:GOSUB2100
1240 FORL=1TOK*10:NEXTL,K
1250 IFPOINT(54,23)=0THENGOSUB2050
1260 GOSUB1840:GOTO860
1270 FORK=2TOLEN(W$)*2STEP2:G$(K)=MID$(W$,K/2,1):NEXTK
1280 GOSUB810:RETURN
1290 GOSUB2050:GOSUB1840

```



```

1300 M$="YOU'VE GOT IT !!":W=1:GOSUB1270
1310 KK=4:LL=50:GOSUB2070
1320 FORK=1T04:GOSUB2100:FORL=1T0K*10:NEXTL
1330 PRINT@88,M$;:FORL=1T0K*5:H=30:GOSUB1800:NEXTL,K
1340 PRINT@298,STRING$(21,32);:IFW=1THEN1460
1350 GOSUB2100:PRINT@84,"THE MYSTERY WORD WAS : ";W$;
1360 FORK=1T0999:NEXTK:GOSUB1270
1370 FORK=1T03:GOSUB2100:FORL=1T030:NEXTL
1380 FORM=1T03:GOSUB2100:FORL=1T03:NEXTL
1390 PRINT@88,"H A N G M A N";:FORL=1T04:NEXTL,M,K
1400 FORK=1T030:NEXTK:GOSUB1890:GOSUB1620
1410 GOSUB2050:GOSUB1840:KK=4:LL=20:GOSUB2070
1420 PRINT@83,"DO YOU WANT ANOTHER GAME ?":I$=INKEY$
1430 LZ=12:JZ=30:KZ=15:T$=T$(3-W):I=0
1440 IZ=IZ+1:GOSUB1760:IFIZ=LEN(T$)-1THENIZ=0
1450 I$=INKEY$:IFI$=""THEN1440ELSEIFI$="N"THENCLS:ENDELSE
GOSUB2100:PRINT@23,"NEW GAME COMING UP":CLEAR2500:GOTO80
1460 PRINT@298," ";
1470 LZ=20:JZ=8:KZ=10:T$=T$(0):IZ=1
1480 FORK=3T08:IFRND(2)=1THEN1520
1490 P=K*64+33:IFK<7THENP=P+1:IFK=30RK=6THENP=P+1
1500 FORI=1T033+RND(5):GOSUB1760:IZ=IZ+1:IFIZ=LEN(T$)THENIZ=1
1510 PRINT@P,MID$(AR$,I,28);:PRINT@630,MID$(WW$,I,8);:NEXT
1520 NEXTK:PRINT@630," ";:GOTO1420
1530 FORK=10T040:SET(K,29):NEXTK:RETURN
1540 FORK=26T028:SET(10,K):SET(23,K):NEXTK
1550 FORK=10T023:SET(K,25):NEXTK:RETURN
1560 FORK=29T09STEP-1:SET(40,K):SET(39,K):NEXTK:RETURN
1570 FORK=38T015STEP-1:SET(K,9):NEXTK
1580 FORK=34T039:SET(K,K/2-7):NEXTK
1590 RETURN
1600 FORK=10T014:SET(15,K):IFK>12SET(16,K)
1610 NEXTK:PRINT@326,S$(0);:PRINT@390,S$(1);:RETURN
1620 RESET(20,25):RESET(19,25):RESET(18,25):SET(20,26)
:SET(19,26):SET(18,26):GOSUB1680
1630 RESET(17,25):RESET(16,25):RESET(20,26):RESET(19,26)
:RESET(18,26):SET(16,26):SET(17,26):SET(18,27):SET(19,27)
:GOSUB1680
1640 RESET(15,25):RESET(14,25):RESET(16,26):RESET(17,26)
:RESET(18,27):RESET(19,27):SET(14,26):SET(15,26)
:SET(16,27):SET(17,28):GOSUB1680
1650 RESET(15,26):RESET(16,27):RESET(17,28):SET(14,27)
:SET(14,28):GOSUB1680
1660 FORK=0T04STEP2:RESET(16,12+K/2):PRINT@326,S$(K);
1670 PRINT@390,S$(K+1);:FORL=1T040:NEXTL,K:RETURN
1680 FORL=1T020:NEXTL:RETURN
1690 M$="YOU LOST !!":W=0:GOTO1310
1700 F=F+1:IFF=14THENF=-1:PRINT@609,CHR$(140);CHR$(140);
1710 IFF=0THENPRINT@609,CHR$(176);CHR$(176);
1720 RD=RND(10):IFRD=1THENII=II-1:IFII<1THENII=2
1730 IFRD=2THENII=II+1:IFII>7THENII=6
1740 IFRD<3THENGOSUB780
1750 RETURN
1760 H=(ASC(MID$(T$,IZ,1))-64)*KZ+1
1770 L=LZ:M=ASC(MID$(T$,IZ+1,1))-47:IFM<11THENL=LZ*M*2:IZ=IZ+1

```

```

1780 IFH>255THENFORZZ=1T0JZ:NEXT:ELSEGOSUB1800
1790 RETURN
1800 ZH=H+256*L:IFZH>32768THENZH=ZH-65536
1810 ZV=VARPTR(M$(0))+1:V1=INT(ZV/256):V=ZV-V1*256
1820 POKE16526,V:POKE16527,V1:X=USR(ZH):RETURN
1830 LL=5:K=6:GOSUB2030:K=8:GOSUB2030:K=4:GOSUB2030:RETURN
1840 LL=30:K=4:GOSUB2030:K=8:GOSUB2030:K=6:GOSUB2030
1850 GOSUB770:RETURN
1860 FORII=7T04STEP-1:GOSUB780:FORL=1T030:NEXTL,II
1870 I=1:FORK=1T08:II=II+I:IFII<>4THENI=-I
1880 GOSUB780:FORL=1T030:NEXTL,K
1890 KK=1:LL=10:GOSUB2070
1900 GOSUB2050:FORI=1T04
1910 KK=14:GOSUB1930:FORL=1T020:NEXTL:KK=21:GOSUB1930
1920 FORL=1T020:NEXTL,I:RETURN
1930 FORK=0T02:POKE15768+K,D(KK+K):NEXTK
1940 FORK=0T03:POKE15832+K,D(KK+K+3):NEXTK
1950 RETURN
1960 KK=4:LL=RND(80):GOSUB2070
1970 GOSUB2050:KK=7:GOSUB1930:L=3:H=100:GOSUB1800:
FORX=53T067:Y=X-29
1980 IFPOINT(X,Y)RESET(X,Y):SET(X,Y):NEXT
1990 SET(X,Y):RESET(X,Y):NEXT
2000 PRINT@865,"*":L=5:H=10:GOSUB1800:L=50:H=20:GOSUB1800:
PRINT@865,CHR$(131);:RETURN
2010 PRINT@405," ";:PRINT@418," ";:PRINT@469," ";
2020 PRINT@482," ";:RETURN
2030 GOSUB2010:PRINT@408,D$(K);:PRINT@472,D$(K+1);:FORL=1TOLL
2040 NEXTL:RETURN
2050 LL=50:K=10:GOSUB2030:KK=0:GOSUB1930:FORL=1T030
2060 NEXTL:RETURN
2070 IK=1:IFKK<II THENIK=-1
2080 IFII<1THENII=1ELSEIFII>7THENII=7
2090 FORII=1TOKKSTEP1K:GOSUB780:FORJ=1TOLL:NEXTJ,II:RETURN
2100 PRINT@0,STRING$(63,32):PRINT@64,STRING$(63,32)
2110 L=30:H=20:GOSUB1800:H=40:GOSUB1800:RETURN
2120 X=RND(150):IFLV$="S"THENX=RND(418)
2130 FORI=1TOX:READW$:NEXTI
2140 FORI=1T02*LEN(W$)STEP2:G$(I)=" ":G$(I+1)=CHR$(95):NEXTI
2150 RETURN
2160 DATAAEROPLANE,CAR,TRUCK,HORSE,CAT,DOG,BICYCLE,HOUSE,HOLIDAY
,SCHOOL,ROAD,DOCTOR,TEACHER,NURSE,ROOM,CLASS,PENCIL,PAPER,BABY,FA
THER,MOTHER,SISTER,BROTHER,FRIEND,RUBBER,ROCKET,CHICKEN,HEN
2170 DATAFARM,COW,HOSPITAL,SHEEP,BUTTER,COLOUR,YELLOW,BEDROOM,BA
TH,FOOTBALL,OCEAN,WATER,WOOL,WEATHER,RAIN,SNOW,SUNSHINE,HOME,SPEA
K,WITCH,WORK,WOMAN,TENNIS,TOP,TEACUP,BREAKFAST,LUNCH,DINNER
2180 DATAFOOT,KNEE,LEG,APPLE,ORANGE,LEMON,QUEEN,KING,CASTLE,ARIT
HMETIC,BACKWARD,BALLOON,BANK,SHOP,ENGLISH,PHOTOGRAPH,BLACKBOARD,B
LOSSOM,BOTTLE,CATERPILLAR,CAMERA,CIRCUS,ELEPHANT,ZEBRA
2190 DATADONKEY,HEADMASTER,COUNTRY,CRICKET,ELECTRICITY,EVENING,E
XPLORE,FAMILY,FLAVOUR,FIFTEEN,HARBOUR,HEART,HELICOPTER,HISTORY,PO
LICEMAN,JOURNEY,KIWI,LAUGH,LESSON,MARBLE,MARSHMALLOW,MISTAKE
2200 DATAMOON,EARTH,MONSTER,MYSTERY,MUSIC,SCIENCE,NATURE,NEEDLE,
NEIGHBOUR,NUMBER,LETTER,HUNDRED,ONION,ORCHESTRA,PIANO,PARROT,PEAC
OCK,DOLLAR,PEOPLE,PICNIC,PICTURE,PILOT,PLANT,PUPPY,QUARTER

```

2210 DATAQUESTION, ANSWER, RABBIT, RADIO, TELEVISION, TELEPHONE, RASPB
 ERRY, SANDWICH, BISCUIT, SCOUT, SEVEN, EIGHT, SHELL, SIMPLE, DAUGHTER, SUB
 TRACT, SUBJECT, SUMMER, AUTUMN, SPRING, HORRIBLE, TICKET, TOMORROW
 2220 SENIOR
 2230 DATAACQUIRE, ACROBAT, ACTIVATE, ADJACENT, ADVOCATE, AESTHETIC, AF
 FLUENT, AFLAME, AGENDA, AGGRAVATE, AGRICULTURE, AISLE, ALARM, ALBATROSS,
 ALGEBRA, BALANCE, BAMBOO, BANANA, BASKET, BATTERY, BIBLIOGRAPHY
 2240 DATACALCULATE, CALENDAR, CAMOUFLAGE, CANDIDATE, CAPACITY, CAPITA
 L, CAPTIVE, CASSEROLE, CATEGORY, CENTENARY, CHAMPION, CHARACTER, CHEMICA
 L, CHOCOLATE, CLASSICAL, CLINIC, COMBINE, DECLARE, DECEIVE, DEGREE
 2250 DATADEMONSTRATE, DEPRECIATE, DESCEND, DEVICE, DIET, DIFFICULTY, D
 ISCREET, DISPLAY, DOCUMENT, EMIGRATE, ENGRAVE, EQUAL, ESTIMATE, EXERCISE
 , FAMOUS, FALSE, FLEXIBLE, FORGIVE, FRAGILE, GALLERY, GENEROUS
 2260 DATAGLANCE, GRATEFUL, GUIDE, HANDICAP, HARMONY, HEIR, HEPTAGON, HO
 LLOW, HONOUR, COMPUTER, HYGIENE, IDLE, ILLUSTRATE, IMPATIENCE, IMPRESS, I
 NCREASE, INDICATE, INNOCENT, INTELLECT, INTRUDE, ITCH, JACKET
 2270 DATAJUNCTION, KALEIDOSCOPE, KNOCK, LANTERN, LEAGUE, LEASH, LEOPAR
 D, LICENCE, LITERAL, LODGE, LOTION, MACHINE, MAINTAIN, MAMMOTH, MANIPULAT
 E, MANUFACTURE, MARINE, MEDIUM, MAXIMUM, MESSAGE, METEOR, MIDDLE
 2280 DATAMILLION, MISCHIEF, MISCELLANEOUS, MOMENT, MULTIPLY, MUSTARD,
 NARROW, NEPHEW, NECESSARY, NEUTRAL, NONSENSE, OFFICE, OPERATION, OPINION
 , OVERHAUL, PADDOCK, PAINTER, PARAGRAPH, PASTE, PATROL, PEASANT
 2290 DATAPENSION, PERMANENT, PERSIST, PHYSICAL, PIECE, PILLOW, PLANET,
 VENUS, MERCURY, MARS, JUPITER, URANUS, NEPTUNE, PLUTO, PLASTER, PLASTIC, P
 LATEAU, PNEUMONIA, POISON, POPULAR, POSITION, POVERTY, PRACTICE
 2300 DATAPROGRAMME, PROVIDE, PUNCTURE, QUANTITY, QUEUE, QUOTE, RADAR, R
 ANK, RATIO, REACT, RECIPROCAL, RECOGNISE, REFER, REFRIGERATE, REHEARSAL,
 RELATIVE, REQUIRE, RESPECT, RHYTHM, RIDICULOUS, ROTATE, RUIN, RUSH
 2310 DATASABOTAGE, SADDLE, SATISFY, SATURN, SCARLET, SCENE, SCISSORS, S
 CRATCH, SCREAM, SECONDARY, SENTIMENT, SEPERATE, SHAFT, SHINGLE, SHOULDER
 , SIGNATURE, SINGLE, SLEEVE, SOCIAL, SOUTH, SPECULATE, SPIRAL, SPLIT
 2320 DATASPORT, SQUEAL, STADIUM, STATUE, STOMACH, STRANGE, SUBSTITUTE,
 SURGEON, SYMPATHY, TACTIC, TALENT, TARNISH, TEMPORARY, THOROUGH, THROAT,
 TIMBER, TYRE, TOUCH, TRAFFIC, TREASURE, TRIANGLE, TROUBLE, TURTLE
 2330 DATAULTIMATE, UNDERSTAND, USUAL, VAGUE, VALUE, VANISH, VARIETY, VE
 RSATILE, VEGETABLE, VOLUME, VISUAL, VOWEL, WAGE, WANDER, WARRANT, WATCH, W
 EIGH, WETHER, WHISPER, WHOLE, WITNESS, WORTH, WRONG, YIELD, YOUTH
 2340 DATAZERO, ZENITH, WALNUT, VIEWER, VELOCITY, TURBULANCE, TORCH, STR
 ENGTH, SPHERE, SILVER, SCORCH, SCHEDULE, REVOLVE, PRESCRIBE, PERIOD, ODD,
 MODIFY, MATERIAL, LIQUID, JUICE, HASTE, FUTURE, EXTREME, DELICATE

** FILES **

1 POKE16561, 250: POKE16562, 253: POKE16544, 200: POKE16545, 253: CLEARS
 00: N=220
 2 Z1\$="E7DF3FFFCDEC6F2A45252424E72B22F003EFE10511001EE72727272E4
 EF3EF400E2FFEBE002EF3221F003021F2F42F4EDCFE7DFCC0002FF702F42F7
 3EFCDC3C33429F234A3DF3DF3DFDCCE0F3DFB20FF0C303DFFFED2DFE5DF117FODC
 F7E5DFDECE2DFA727275EC55433300000"

3 Z2\$="D3DE1DE555DDEA40E363E36BE36B0233A0E560F91A09B133231303D3
 3EA0EF60B1FED0B6520E6030B636030B2902B02D0111DBDE3C6621FEE43E009B
 20E99E323016E240F29E2AE28E99D335ABE70A1ED0E12BE515511EDB9E93BE744
 EED39E119518EF73737F1925E300D0000"
 4 CLS:PRINT@532,"I N I T I A L I S I N G":FORI=1TON:PRINT@606,N-
 I;" :N1=ASC(MID\$(Z1\$,I,1)):GOSUB5:I1=N1:N1=ASC(MID\$(Z2\$,I,1)):
 GOSUB5:POKEI-513,16*I1+N1:NEXTI:GOTO6
 5 IFN1>57THENN1=N1-55:RETURN:ELSEN1=N1-48:RETURN
 6 Z1\$="":Z2\$="":CLEAR50:POKE16526,126:POKE16527,254:X=USR(0)
 20 REM *** DATA PROCESSING AND FILING SYSTEM ***
 30 ' *** BY EDWIN R. PAAY 39 FAIRVIEW GRV. HACKHAM WEST 5163 ***
 40 CLS:PRINT"
 --- SELECT REQUIRED FUNCTION ---
 50 PRINT"
 1 ==> ENTER NEW DATA TO FILE.
 2 ==> SEARCH FILE FOR DATA.
 3 ==> EXIT FROM PROGRAM.
 4 ==> EDIT FILE.
 5 ==> LIST ALL DATA.
 60 PRINT"
 ":A=0:FG=0:H=0:PRINT@780,"";
 70 A\$=INKEY\$:A=A+1:IFA\$<>"GOTO110
 80 GOSUB250 :GOTO70
 110 IFA\$>"6"ORA\$<"1"THEN70 ELSEB=VAL(A\$):ONB6GOTO120 ,310 ,54
 0 ,560 ,670
 120 ' ** ENTER NEW DATA **
 130 CLS:PRINT"TYPE IN DATA SEPARATED WITH COMMA'S , HIT <ENTER>
 AFTER EACH GROUP OF DATA, THEN TYPE <ENTER> AGAIN WHEN ";CHR\$(
 34);"READY";CHR\$(34);" APPEARS ON THE SCREEN.
 140 PRINT"EXAMPLE : DATA , DATA , DATA , DATA <ENTER> - <ENTER
 >
 TYPE ";CHR\$(34);CHR\$(92);CHR\$(34);" TO STOP ANYTIME WHILE CURSO
 R IS BLINKING !
 * * * ";MEM;" BYTES FREE. * * *
 ----- READY TO ACCEPT DATA ! ! ! -----
 150 A=2:BUFFER=-257:POKEBU,136:BU=BU+1:POKEBU,34:PRINTCHR\$(34);:
 BU=BU+1:E=0
 160 GOSUB250 :A\$=INKEY\$:IFA\$=" "THEN160
 170 IFA\$>250THENCLS:PRINT" *** ERROR *** ERROR *** ERROR
 *** ERROR ***
 ----- DATA FIELD TO LONG CANNOT ENTER LAST CHARACTER !!! -----
 ":ELSEGOTO190
 180 FORC=0TO3000:NEXT:GOSUB280 :GOTO130
 190 PRINTA\$;:GOSUB250 :A=A+1:IFA\$=","THENPOKEBU,34:BU=BU+1:POKE
 BU,44:BU=BU+1:POKEBU,34:BU=BU+1:PRINTCHR\$(8);CHR\$(34);";";CHR\$(34
);:A=A+3:GOTO160
 200 IFA\$=CHR\$(10)GOTO40

```

210 IFA%=CHR$(13) THENPOKEBU, 34:BU=BU+1:POKEBU, 44:BU=BU+1:POKEBU,
34:BU=BU+1:POKEBU, ASC("@"):BU=BU+1:POKEBU, 34:BU=BU+1:POKEBU, 44:A=
A+6:GOTO280
220 IFA%=CHR$(8) THENBU=BU-1:PRINTCHR$(30);:GOTO160
230 POKEBU, ASC(A%):BU=BU+1:GOTO160
240 ' *CURSOR*
250 D=PEEK(16416)+PEEK(16417)*256:E=E+1
260 IFE>6AND(PEEK(D)=95ORPEEK(D)=32) THENPOKED, 136:E=0:RETURN
270 IFE>6POKED, 32:E=0:RETURN
275 RETURN
280 ' *END OF LINE*
290 POKE16405, 0:POKE16526, 0:POKE16527, 254:X=USR(0)
300 POKE16405, 1:NU=PEEK(-285)+PEEK(-284)*256:NU%=STR$(NU)
302 X=PEEK(-288)+PEEK(-287)*256:FORG=2TOLEN(NU%):Y=X+G-2:Y=Y+655
36*(Y>32767):POKEY, ASC(MID$(NU%, G, 1)):NEXT
305 GOTO130
310 ' ** SEARCH **
320 RESTORE:CLS:READLE$:LE=VAL(LE$):DIMB$(LE):DIMC$(LE):ONERRORG
OTO432 :B$="":FL=0:Z$=""
325 FORJ=1TOLE:READC$(J):NEXT:GOSUB460 :READW
330 PRINT"
TYPE ";CHR$(34);CHR$(92);CHR$(34);" TO STOP ! ! ! .
----- ENTER DATA TO BE SEARCHED FOR. -----
"
340 GOSUB250 :A%=INKEY%:IFA$=""GOTO340
342 PRINTA%:IFA%=CHR$(10)THEN40 ELSEIFA%=CHR$(8)THEN344 ELSE
IFA%<>CHR$(13) THENB%=B%+A%:GOTO340
343 GOTO350
344 GOSUB660 :PRINTCHR$(30);:GOTO340
350 M=0:FORJ%=1TOLE:READB$(J%):GOSUB800:M=M+SS:NEXTJ%:READH$:REA
DH
355 IFM=0THEN350 ELSEFL=1
410 CLS:FORJ=1TOLE:PRINTC$(J);" ";B$(J):NEXT:PRINT"DATA IS IN L
INE ";H;
412 IFFG=1THENRETURN
420 PRINT" <<< >>> CONTINUE SEARCH OR STOP (C/S)";
421 A%=INKEY%:GOSUB250 :IFA$=""THEN421 ELSEIFA$="C"THEN350
422 GOTO40
432 IF(ERR/2)+1=10RESUMENEXT
435 IF(ERR/2)+1 <> 4THEN450
440 IFERL=350 THENRESUME508
450 PRINT"*** ERROR ";(ERR/2)+1;" IN LINE ";ERL;" ***":END
460 READC$:IFC$="@" THENRETURNELSE460
508 CLS:PRINTCHR$(23)
520 IFFL=1THENPRINT"
** ALL DATA HAS BEEN SEARCHED **:FORJ=0TO1500:NEXT:GOTO320
530 PRINT"
* * * DATA NOT FOUND ! ! * * *":FORJ=0TO1500:NEXT:GOTO320

```

```

540 CLS:END
550 '*** EDIT FILE ***
560 CLS:PRINT"TO DELETE OR EDIT FILES STOP THE PROGRAM AND EDIT
DATA LINES USING THE EDIT COMMAND PROVIDED BY BASIC.
IF YOU WISH YOU CAN ENTER DATA AND I WILL TELL YOU WHICH LINES
570 PRINT"CONTAIN THE DATA IN QUESTION.
----- ENTER DATA OR TYPE < ";CHR$(92);" > -----
"
580 B$="":ONERRORGOTO630
590 GOSUB250 :A%=INKEY%:PRINTA%:IFA%=CHR$(10)THEN40 ELSEIFA%
=CHR$(13)THEN600 ELSEIFA%=CHR$(8)THENGOSUB660 :PRINTCHR$(30);:G
OTO590 ELSEB%=B%+A%:GOTO590
600 RESTORE:GOSUB460 :READX
610 READC$:IFC%=B%THEN620 ELSE610
620 GOSUB460 :READX:PRINTX;:GOTO610
630 IFERL=610 AND(ERR/2)+1=4THENRESUME650
640 GOTO450
650 PRINT"
*** SEARCH COMPLETED ***":INPUT"
TYPE <ENTER> TO CONT. ";A%:GOTO560
660 B%=LEFT$(B%, (LEN(B%)-1)):RETURN
670 '*** LIST ALL ***
680 RESTORE:ONERRORGOTO730 :READLE$:LE=VAL(LE$):DIMB$(LE):DIMC$
(LE):FG=1
690 FORJ=1TOLE:READC$(J):NEXT:GOSUB460 :READX
700 CLS:GOSUB751 :READA%:READX:PRINTCHR$(8);CHR$(8);X;" CONTINU
E (Y/N) ?";
710 A%=INKEY%:GOSUB250 :IFA$=""GOTO710
720 IFA$="Y"THEN700 ELSE40
730 IFERL=751 AND(ERR/2)+1=4RESUME40
740 IF(ERR/2)+1=10RESUMENEXT
750 GOTO450
751 FORJ=1TOLE:READB$(J):NEXTJ:GOTO410
800 FORSS=1TOLEN(B$(J%))-LEN(B%)+1
805 IFB%=MID$(B$(J%), SS, LEN(B%))RETURN
810 NEXT:SS=0:RETURN
998 REM*** DATA STARTS HERE ***
1000 DATA"4","MEMORY LOCATION :","FUNCTION :","CALL SEQUE
NCE :","COMMENTS :","@",1000
1010 DATA"0000-01D8","SYSTEM INIT. I/O SUBR.,"N.A.",",", "@",1010
1020 DATA"01D9-03E2","CASSETTE SUBR.,"N.A.",",", "@",1020
1030 DATA"03E3-0457","KEYBOARD DRIVER","N.A.",",", "@",1030
1040 DATA"0458-058C","VIDEO DRIVER","N.A.",",", "@",1040
1050 DATA"058D-673","LPRINT DRIVER","N.A.",",", "@",1050
1060 DATA"0674-070A","INITIALIZE","N.A.",",", "@",1060
1070 DATA"070B-1607","FLOATING POINT MATH","N.A.",",", "@",1070
1080 DATA"1608-164F","TABLE LEVII ENTRY POINTS","N.A.",",", "@",10
80
1090 DATA"1650-1820","TABLE BASIC COMMANDS",",",", "@",1090
1100 DATA"1821-191C","TABLE OF JUMP ADRS. FOR BASIC COMMANDS.",
N.A.",",", "@",1100

```

```

1110 DATA"3000-37DD","RESERVED FOR DMA DEVICES","N.A.,"THIS SPA
CE IS FREE AND CAN BE USED","@",1110
1120 DATA"37DE","DOS COMMUNICATION ADRS","",",",",@",1120
1122 DATA"3C00-3FFF","VIDEO DISPLAY","N/A","N/A","@",1122

10 'DATABASE VERSION 4.1 10/81
(C) GRAEME MOAD
6/278 DOMAIN ROAD
SOUTH YARRA VIC 3141
20 CLS: CLEAR2000: DEFSTRA-E: DEFINTF-Z: ONERRORGOTO1370
30 DIMER(11): FORI=0TO11: READER(I): NEXT: FORI=0TO10: READAP(I): NEXT
40 DATA, TOTAL FIELD LENGTHS <255, BAD RECORD NUMBER, BAD FIELD NAM
E/NUMBER, FILE TOO LARGE, FILE INCOMPATIBLE, FILE NOT FOUND, DISK FUL
L, BAD FILE NAME, DISK WRITE PROTECTED OR NOT AVAILABLE, BAD PARAMET
ER, NO MORE SPACE
50 DATAPAGE LENGTH, LINE LENGTH, TOP MARGIN, BOTTOM MARGIN, LEFT MAR
GIN, LINES/RECORD, LINES BETWEEN RECORDS, RECORDS/PAGE, MAX RECORD WI
DTH, COLUMNS BETWEEN RECORDS, RECORDS/LINE
60 AF(9)="STRING": AF(10)="INTEGER": SK=0
70 D0=STRING$(255,128): D1="0123456789": D2="ABCDEFGHIJKLMNPOQRSTU
VWXYZabcdefghijklmnopqrstuvwxyz !#$%&'()*+,-./:<>?": D3="-,:."
80 D4=CHR$(30): D5=CHR$(13): D9=CHR$(136): GOTO360
90 GOSUB100: GOSUB110: M=INSTR(D,B) OR INSTR(DA,B): IFMTHENPRINT@896,
D4: DA=D9: RETURNELSE90
100 PRINT@896,E"? "D9D4: PRINT@960,ER(IE);: IE=0: RETURN
110 B=INKEY$: IFB=""THEN110ELSERETURN
120 E="PRESS ENTER TO CONTINUE": D=D5: GOSUB90: RETURN
130 E="IS THIS OK"
140 E=E+" <Y/N>": D="YN": DA="yn": GOSUB90: RETURN
150 LSETBS="": GOSUB100: S1=1
160 ML=16258+LEN(E): POKEVARPTR(BS)+2, INT(ML/256): POKEVARPTR(BS)+
1, ML-INT(ML/256)*256: POKEVARPTR(BS), S3
170 POKE16257+LEN(E)+S1, 136+(S1>S3)*76
180 GOSUB110: S2=ASC(B)
190 IF INSTR(D,B) AND S1<=S3 POKE16257+LEN(E)+S1, S2: S1=S1+1
200 IFS2=8 AND S1>1 S1=S1-1: POKE16258+LEN(E)+S1, 32
210 IFS2=13 BS=MID$(BS,1,S1-1): PRINT@896, D4: S4=VAL(BS): RETURN
220 IFS2=24 THEN150 ELSE170
230 GOSUB150: X=S4-1-(S4=0)
240 S=1: GOSUB270: Y=VAL(MID$(BS,T+1))-1: IFY<0Y=NR-1
250 IFX<0ORX>NR-1ORY<0ORY>NR-1IE=2
260 E=E+STR$(X+1)+" "+STR$(Y+1): W=Y-X+1: RETURN
270 FORI=1TO4: T=INSTR(S,BS,MID$(D3,I,1)): IFT=ONEXT: RETURNELSERET
URN
280 FL=0: FORI=0TONF-1: F(2,I)=FL+1: FL=FL+F(1,I): NEXT: IFFL>255THEN
IE=1: RETURNELSEMR=23552/FL: IFMR>500MR=500
290 DIMAR(MR-1): FORI=0TOMR-1: M!=40960+I*FL: POKEVARPTR(AR(I))+2, I
NT(M!/256): POKEVARPTR(AR(I))+1, M!-INT(M!/256)*256: POKEVARPTR(AR(I
)), FL: LSETAR(I)=D0: PRINT@50, I-NR+1: NEXT
300 DIMDS(15): FORI=0TO15: ML=15360+I*64: POKEVARPTR(DS(I))+2, INT(M
L/256): POKEVARPTR(DS(I))+1, ML-INT(ML/256)*256: POKEVARPTR(DS(I)), 6
4: NEXT

```

```

310 DIMDU(NF-1): FORI=0TONF-1: ML=15632+I*64: POKEVARPTR(DU(I))+2, I
NT(ML/256): POKEVARPTR(DU(I))+1, ML-INT(ML/256)*256: POKEVARPTR(DU(I
)), 48: NEXT
320 DIMDT(NF): F1=4: FORI=0TONF-1: F2=F(1,I)*(60-NF)/FL: M!=64512+F1
: GOSUB330: F1=F1+F2+1: NEXT: M!=64512: F2=64
330 POKEVARPTR(DT(I))+2, INT(M!/256): POKEVARPTR(DT(I))+1, M!-INT(M
!/256)*256: POKEVARPTR(DT(I)), F2: RETURN
340 DIMF(6,NF-1), BL(16), BM(NF-1,3): FORI=0TONF-1: F(3,I)=I: NEXT: RE
TURN
350 FORI=SKTOOSTEP-1: CMD"D", NR, AR(0)(F(2,F(3,I)), F(1,F(3,I))): NE
XT: RETURN
360 CLS: PRINT@0,"DATABASE MANAGEMENT SYSTEM VERSION 4.1 10/81
MEMORY SIZE SHOULD BE SET AT 40960"
370 PRINT@256,"1 - INITIALIZE NEW FILE
2 - LOAD FILE FROM DISK": E="OPTION": D="12": GOSUB90
380 ONMGOSUB1150,1260: IFIE<>0THEN360ELSE390
390 CLS: GOSUB1430: PRINT@128,"A - ADD", "D - DELETE
E - EDIT", "F - FIND
L - LIST", "M - MERGE
N - NEW", "P - PRINT
S - SORT", "W - WRITE": E="OPTION": D="ADEFMLNPSW": DA="adeflmnpsw"
400 GOSUB90: ONMGOSUB410,470,450,510,620,1290,690,700,1040,1210: G
OTO390
410 CLS: C1="ADD / NEXT / PREVIOUS": C2="NPA": C3="npa": GOSUB420: GO
TO630
420 IFNR+1>MRTHENIE=1: RETURNELSENR=NR+1: R=NR: LSETAR(NR-1)="": GO
SUB1430: GOSUB1460: GOSUB430: RETURN
430 D=D1+D2: FORJ=0TONF-1: E=AF(J): S3=F(1,J): GOSUB150: IE=0: IFBS<>"
"LSETDU(J)=BS: IFF(0,J)RSETDU(J)=BS
440 MID$(AR(R-1),F(2,J))=DU(J): NEXT: RETURN
450 E="EDIT RECORD": S3=3: D=D1: GOSUB230: IFIE<>0RETURNELSER=X+1: GO
SUB1460
460 GOSUB130: IFM=2THENGOSUB430: GOTO460ELSERETURN
470 E="DELETE RECORD(S)": S3=7: D=D1+D3: GOSUB230: IFIE<>0RETURN
480 IFW=1R=X+1: GOSUB1460: E="DELETE THIS RECORD"
490 GOSUB140: IFM=2RETURN
500 FORI=XTOY: LSETAR(I)=D0: NEXT: GOSUB350: NR=NR-W: RETURN
510 E="SEARCH FIELD": S3=16: D=D1+D2: GOSUB150: U=S4
520 IFU<1ORU>NF: FORI=1TONF: IF INSTR(AF(I-1),BS)=1THENU=IELSENEXT:
IE=3: RETURN
530 E="STRING TO FIND": S3=F(1,U-1): GOSUB150: IE=0: F2=1: AS=BS: IFIN
STR(BS,"*")=1F2=F(1,U-1): AS=MID$(BS,2)
540 E="SHORT OR LONG FORMAT": D="SL": DA="s1": GOSUB90: F1=0: CLS: GOS
UB600: IFM=2THEN570
550 E="CONTINUE": F4=1: FORR=1TONR: PRINT@849,R;: F3=INSTR(MID$(AR(R
-1),F(2,U-1),F(1,U-1)),AS): IFF3>0ANDF3<=F2THENF1=F1+1: PRINT@860,F
1;: GOSUB610: IFF1=INT(F1/10)*10THENE="CONTINUE": GOSUB140: IFM=2RETN
RNEELSECLS: GOSUB600
560 NEXT: GOSUB120: RETURN
570 C1="FIND / NEXT / PREVIOUS": C2="NP F": C3="np f": R=MR: GOSUB14
60: R=0: GOSUB580: GOTO630
580 R=R+1: PRINT@849,R;: F3=INSTR(MID$(AR(R-1),F(2,U-1),F(1,U-1)),
AS): IFF3>0ANDF3<=F2THENF1=F1+1: PRINT@860,F1; ELSEIFR<=NRANDINKEY*
">"Q"THEN580

```

```

590 RETURN
600 GOSUB1430:PRINT@B32,"RECORDS SEARCHED:      FOUND:      STRING
: "AS:RETURN
610 LSETDT(NF)=MID$(STR$(R),2):FORI=0TONF-1:LSETDT(I)=MID$(AR(R-
1),F(2,I),F(1,I)):NEXTI:LSETDS(F4+1)=DT(NF):F4=F4+1:IFF4>10THENF4
=1:RETURNELSERETURN
620 E="LIST RECORD":S3=3:D=D1:GOSUB230:C1="NEXT / PREVIOUS":C2="
NP":C3="np":IFIE<>0RETURNELSER=X+1:GOSUB1460
630 IFR<1ORR>NRRETURNELSEGOSUB1470:GOSUB680:ONMGOSUB430,660,670,
640,650,420,580:GOTO630
640 R=R+1:RETURN
650 R=R-1:RETURN
660 X=R-1:Y=R-1:W=1:GOSUB480:GOSUB1430:RETURN
670 R=0:RETURN
680 E=C1+" / EDIT / DELETE / QUIT":D="EDQ"+C2:DA="edq"+C3:GOSUB9
0:RETURN
690 E="NEW FILE":GOSUB140:IFM=2RETURNELSE20
700 E="PRINT RECORD(S)":S3=7:D=D1+D3:GOSUB230:IFIE<>0RETURNELSEG
OSUB140:IFM=2RETURN
710 E="OUTPUT TO PRINTER OR VIDEO":D="VP":DA="vp":GOSUB90:IFM=1T
HENP2=1:PL=16:PW=64ELSEP2=0:PL=66:PW=80
720 IFPP(0)=OPP(0)=PL:PP(1)=PW:FORI=0TONF-1:F(4,I)=I+1:F(5,I)=0:
F(6,I)=F(1,I):NEXT:PP(2)=0:PP(4)=0:PP(5)=NF:PP(6)=0:PP(7)=PL/NF:P
P(8)=PW:PP(9)=0:PP(10)=1:PP(3)=PP(0)-PP(2)-PP(5)*PP(7)
730 CLS:GOTO890
740 I=0:PP(0)=PL:GOSUB1030:IFPP(0)<1ORPP(0)>PLTHEN740ELSEPP(5)=N
F:PP(7)=PP(0)/NF:PP(3)=PP(0)-PP(2)-PP(5)*PP(7):GOSUB1000
750 I=1:PP(1)=PW:GOSUB1030:IFPP(1)<1ORPP(1)>PWTHEN750ELSEPP(8)=P
P(1):PP(10)=1:GOSUB980:GOSUB1000
760 I=2:GOSUB1030:IFPP(2)>PP(0)THEN760ELSEPP(7)=(PP(0)-PP(2))/NF
:PP(3)=PP(0)-PP(2)-PP(5)*PP(7):GOSUB1000
770 I=4:GOSUB1030:IFPP(4)>PP(1)THEN770ELSEPP(8)=PP(1)-PP(4):GOSU
B980:GOSUB1000
780 I=5:GOSUB1030:IFPP(5)>16ORPP(5)>PP(0)-PP(2)THEN780ELSEPP(7)=
(PP(0)-PP(2))/PP(5):PP(3)=PP(0)-PP(2)-PP(5)*PP(7):FORI=0TONF-1:F(
4,I)=F(4,I)*-(F(4,I)<=PP(5)):NEXT:GOSUB1000
790 I=6:GOSUB1030:IFPP(6)>PP(0)-PP(2)-PP(5)THEN790ELSEPP(7)=(PP(
0)-PP(2))/(PP(5)+PP(6)):PP(3)=PP(0)-PP(2)-(PP(5)+PP(6))*PP(7)+PP(
6):GOSUB1000
800 I=7:GOSUB1030:IFPP(7)>(PP(0)-PP(2))/(PP(5)+PP(6))THEN800ELSE
PP(3)=PP(0)-PP(2)-(PP(5)+PP(6))*PP(7)+PP(6):GOSUB1000
810 I=8:GOSUB1030:IFPP(8)>PP(1)-PP(4)THEN810ELSEGOSUB980:GOSUB10
00
820 I=9:GOSUB1030:IFPP(8)+PP(9)>PP(1)-PP(4)THEN820ELSEGOSUB1000
830 I=10:GOSUB1030:IFPP(10)>4ORPP(10)>(PP(1)-PP(4))/(PP(8)+PP(9)
)THEN830ELSEGOSUB1000
840 IFPP(5)*PP(1)>1024THENIE=10:GOTO740
850 FORI=0TOPP(5)
860 E="FIELD(S) APPEARING ON LINE"+STR$(I):D=D1+D5:GOSUB90:Z=M-2
:IFZ<0ORZ>NF-1NEXT:RETURN
870 F(4,Z)=I:IFITHERE="POSITION":D=D1+D5:S3=2:GOSUB150:IFS4>PP(8
)THEN870ELSEF(5,Z)=S4ELSEGOSUB1000:GOTO860
880 E="WIDTH":S3=2:GOSUB150:IFS4+F(5,Z)>PP(8)THEN870ELSEF(6,Z)=S
4:GOSUB1000:GOTO860

```

```

890 GOSUB1000:GOSUB130:IFM=2THENFORI=0TO10:PP(I)=0:NEXT:GOSUB100
0:GOSUB740:GOTO890
900 FORI=0TOPP(5)-1:M!=64512+I*PP(1):POKEVARPTR(BL(I))+2,INT(M!/
256):POKEVARPTR(BL(I))+1,M!-INT(M!/256)*256:POKEVARPTR(BL(I)),PP(
1):LSETBL(I)="":NEXT
910 FORI=0TONF-1:FORJ=0TOPP(10):M!=64512+(F(4,I)-1)*PP(1)+PP(4)+
J*(PP(8)+PP(9))+F(5,I):POKEVARPTR(BM(I,J))+2,INT(M!/256):POKEVARP
TR(BM(I,J))+1,M!-INT(M!/256)*256:POKEVARPTR(BM(I,J)),F(6,I):NEXTJ
,I:CLS
920 P1=0:IFPP(2)FORI=1TOPP(2):GOSUB970:NEXT
930 FORH=0TOPP(7)-1:IFINKEY$="Q"ORINKEY$="q"THENRETURNELSEGOSUB9
60:NEXTH:IFPP(3)-PP(6)FORI=1TOPP(3)-PP(6):GOSUB970:NEXT
940 PRINT@959,"*":GOSUB110:IFB="Q"ORB="q"RETURNELSEP1=0
950 IFX>YRETURNELSE920
960 P3=1:FORI=0TONF-1:FORJ=0TOPP(10)-1:LSETBM(I,J)=MID$(AR(X+J),
F(2,I),F(1,I))*-(X+J<=Y)):NEXTJ,I:X=X+PP(10):FORI=0TOPP(5)-1:LSET
S(P1)=BL(I):GOSUB970:NEXT:P3=0:IFPP(6)THENFORI=1TOPP(6):GOSUB970:
NEXT:RETURNELSERETURN
970 P1=-P1*(P1<15)+1:IFP2=0ANDP3=1THENLPRINTBL(I):RETURNELSEIFP2
=0THENLPRINT:RETURNELSERETURN
980 FORI=0TONF-1:IFF(6,I)+F(5,I)>PP(8)THENF(6,I)=PP(8)-F(5,I):IF
F(6,I)<0THENF(4,I)=0:F(5,I)=0:F(6,I)=0
990 NEXT:RETURN
1000 PRINT@0,"PARAMETER"TAB(23)"VALUE"TAB(30)"# FIELD NAME"TAB(
44)"LINE"TAB(49)"POSN"TAB(54)"WDTH"TAB(59)"LNTH"
1010 FORJ=0TONF-1:PRINTAP(J)TAB(23)PP(J)TAB(29)J+1TAB(33)AF(J)TA
B(44)F(4,J)TAB(49)F(5,J)TAB(54)F(6,J)TAB(59)F(1,J):NEXT:IFNF<8THE
NFORJ=NFTO10:PRINTAP(J)TAB(23)PP(J):NEXT
1020 RETURN
1030 E=AP(I):D=D1:S3=2:GOSUB150:IFBS<>" THENPP(I)=S4:RETURNELSER
ETURN
1040 CLS:GOSUB1430:PRINT@128,"ORDER # FIELD NAME"
1050 FORI=0TONF-1:PRINT@192+I*64,(I+1)*-(I<=SK)TAB(6)F(3,I)+1TAB
(10)AF(F(3,I)):NEXT:GOSUB130:IFM=1THENGOSUB350:RETURN
1060 E="SORT FIELD(S)"
1070 S3=17:D=D1+D3:GOSUB150:F(3,0)=S4-1:S=1
1080 FORK=1TONF-1:GOSUB270:IFT>0THENF(3,K)=VAL(MID$(BS,T+1))-1:S
=T+1:NEXT
1090 SK=K-1:FORI=0TOSK:IFF(3,I)<0ORF(3,I)>NF-1THENIE=3:GOTO1140
1100 FORJ=0TOSK:IFF(3,I)=F(3,J)ANDI<>JTHENIE=3:GOTO1140ELSENEXTJ
,I
1110 T=SK:FORI=0TONF-1
1120 FORJ=0TOT:IFF(3,J)=I:NEXTIELSENEXTJ:T=T+1:F(3,T)=I:NEXTI
1130 GOTO1040
1140 FORI=0TONF-1:F(3,I)=I:NEXT:GOTO1040
1150 GOSUB1330:AN=BS:E="NUMBER OF FIELDS (<9)":D=D1:GOSUB90:NF=M
-1:IFNF=0THEN20ELSEGOSUB340
1160 GOSUB1440:FORI=0TONF-1:AF(I)="" :F(1,I)=20
1170 GOSUB1450:D=D1+D2:E="FIELD NAME (<=16 CHARACTERS)":S3=16:GO
SUB150:IE=0:AF(I)=LEFT$(BS,16)
1180 GOSUB1450:E="FIELD TYPE (STRING/INTEGER)":D="SI"+D5:DA="si"
+D5:GOSUB90:IFM=3THENM=1
1190 F(0,I)=M-1:GOSUB1450:E="FIELD LENGTH (<=48 CHARACTERS)":D
=D1:S3=2:GOSUB150:N=S4:IFN>48THENIE=10:GOTO1190ELSEIFN>0THENF(1,I
)=N

```

```

1200 GOSUB1450:NEXTI:GOSUB130:IFM=1:GOSUB280:RETURNELSE20
1210 E="WRITE RECORD(S)":S3=7:D=D1+D3:GOSUB230:IFIE<>0RETURNELSE
GOSUB140:IFM=2RETURN
1220 GOSUB1330:GOSUB1340:OPEN"0",2,AI
1230 PRINT#2,W,NF:FORI=0TONF-1:PRINT#2,CHR$(34);AF(I);CHR$(34);F
(0,I);F(1,I);F(4,I);F(5,I);F(6,I):NEXT:FORI=0TO10:PRINT#2,PP(I):N
EXT
1240 OPEN"R",1,AR:GOSUB1360:FORI=XTOYSTEPF:FORJ=0TOF-1:IFI+J<=YT
HENLSETA(J)=AR(I+J)ELSELSETA(J)=""
1250 NEXTJ:PUT1,(I-X)/F+1:NEXTI:CLOSE:RETURN
1260 GOSUB1330:AN=BS:GOSUB1340:OPEN"I",2,AI
1270 INPUT#2,NR,NF:GOSUB1440:GOSUB340:FORI=0TONF-1:INPUT#2,AF(I)
,F(0,I),F(1,I),F(4,I),F(5,I),F(6,I):GOSUB1450:NEXT:FORI=0TO10:INP
UT#2,PP(I):NEXT:GOSUB280:IFNR>MRTHENIE=4:GOTO1320
1280 OPEN"R",1,AR:GOSUB1360:FORI=0TONR-1STEPF:FORJ=0TOF-1:GET1,I
/F+1:LSETAR(I+J)=A(J):NEXTJ,I:CLOSE:RETURN
1290 GOSUB1330:GOSUB1340:OPEN"I",2,AI:INPUT#2,NO,N1:IFN1<>NFTHEN
IE=5:GOTO1320ELSEIFNR+NO>MRTHENIE=4:GOTO1320
1300 FORI=0TONF-1:INPUT#2,B,N1,N2:IFN1<>F(0,I)ORN2<>F(1,I)THENIE
=5:GOTO1320ELSEINPUT#2,N1,N1,N1:NEXT
1310 OPEN"R",1,AR:GOSUB1360:FORI=0TONO-1STEPF:FORJ=0TOF-1:GET1,I
/F+1:LSETAR(NR+I+J)=A(J):NEXTJ,I:NR=NR+NO
1320 CLOSE:RETURN
1330 E="FILE NAME (<8 CHARACTERS)":S3=8:D=D1+D2:GOSUB150:IE=0:AR
=BS:AI=BS:RETURN
1340 E="DRIVE NO. (0-3)":S3=1:D=D1:GOSUB150:IFBS<>" "THENBS="":+B
S
1350 AR=AR+"/TXT"+BS:AI=AI+"/DAT"+BS:RETURN
1360 F=256/FL:FORI=0TOF-1:FIELD1,(I*FL)ASA,(FL)ASA(I):NEXT:RETUR
N
1370 IFERR/2=53THENIE=6:RESUME1320
1380 IFERR/2=61THENIE=7:RESUME1320
1390 IFERR/2=57ORERR/2=64ORERR/2=69THENIE=8:RESUME1320
1400 IFERR/2=67ORERR/2=68THENIE=9:RESUME1320
1410 IFERR/2=5THENIE=10:RESUME1320
1420 ONERRORGOTO20
1430 PRINT#0,"FILE: "ANTAB(18)"FIELDS:"STR$(NF)TAB(30)"RECORDS:"
STR$(NR)TAB(44)"SPACE:"STR$(MR-NR)D4:RETURN
1440 CLS:GOSUB1430:PRINT#128," # "TAB(4)"FIELD NAME"TAB(22)"TYPE"
TAB(30)"LENGTH":RETURN
1450 PRINT#192+I*64,I+1;TAB(4)AF(I)TAB(21)AF(F(0,I)+9)TAB(31);F(
1,I):RETURN
1460 PRINT#128,"RECORD:"D4:FORI=0TONF-1:LSETDS(I+4)=AF(I):NEXT
1470 PRINT#135,R" ";:FORI=0TONF-1:LSETDU(I)=MID$(AR(R-1),F(2,I)
,F(1,I)):NEXT:RETURN

```

** DUPLEX **

```

42E9      00100 START  EQU   42E9H
09D7      00110 MOVSTR EQU   09D7H
2337      00120 EVAL   EQU   2337H

```

```

0A7F      00130 SHAPE  EQU   0A7FH
1997      00140 SNERR  EQU   1997H
260D      00150 VARPT  EQU   260DH
0890      00160 RETN   EQU   0890H
415B      00200 ORG    415BH
415B C39B43 00210 JP     DEF
41A9      00220 ORG    41A9H
41A9 C3C643 00230 JP     USR
42E9      00300 ORG    START
42E9 00     00310 TABLE DEFB  0
0064      00320 REST   DEFS  100
434E EA42   00330 TNEXT  DEFW  REST
42A0      00340 STORE  EQU   42A0H
4350 DD7300 00400 DIXIE  LD    (IX),E
4353 DD23   00410 INC    IX
4355 DD7200 00420 LD    (IX),D
4358 DD23   00430 INC    IX
435A C9     00440 RET
435B 2B     00450 SCAN  DEC   HL
435C D7     00460 RST    16
435D E5     00470 PUSH   HL
435E 0E00   00480 LD    C,0
4360 0C     00490 LP1   INC   C
4361 23     00500 INC    HL
4362 7E     00510 LD    A,(HL)
4363 FE20   00520 CP    32
4365 2808   00530 JR    Z,NXT1
4367 FED5   00540 CP    213
4369 2804   00550 JR    Z,NXT1
436B FE28   00560 CP    40
436D 20F1   00570 JR    NZ,LP1
436F E1     00580 NXT1  POP   HL
4370 11E942 00590 LD    DE, TABLE
4373 E5     00600 LP2   PUSH  HL
4374 D5     00610 PUSH  DE
4375 41     00620 LD    B,C
4376 1A     00630 LP3   LD    A,(DE)
4377 BE     00640 CP    (HL)
4378 2012   00650 JR    NZ,NXT2
437A 23     00660 INC   HL
437B 13     00670 INC   DE
437C 10F8   00680 DJNZ  LP3
437E D5     00690 PUSH  DE
437F DDE1   00700 POP   IX
4381 DD7E02 00710 LD    A,(IX+2)
4384 B7     00720 OR    A
4385 2005   00730 JR    NZ,NXT2
4387 D1     00740 POP   DE
4388 E1     00750 POP   HL
4389 3EFF   00760 LD    A,255
438B C9     00770 RET
438C D1     00780 NXT2  POP   DE
438D 1A     00790 LP4   LD    A,(DE)
438E 13     00800 INC   DE
438F B7     00810 OR    A

```

4390	20FB	00820	JR	NZ,LP4
4392	2A4E43	00830	LD	HL, (TNEXT)
4395	DF	00840	RST	24
4396	E1	00850	POP	HL
4397	20DA	00860	JR	NZ,LP2
4399	AF	00870	XOR	A
439A	C9	00880	RET	
439B	CD5B43	01000	DEF CALL	SCAN
439E	F5	01010	PUSH	AF
439F	41	01020	LD	B,C
43A0	EB	01030	EX	DE,HL
43A1	CDD709	01040	CALL	MOVSTR
43A4	EB	01050	EX	DE,HL
43A5	D5	01060	PUSH	DE
43A6	2B	01070	DEC	HL
43A7	D7	01080	RST	16
43A8	CF	01090	RST	8
43A9	D5	01100	DEFB	213
43AA	CD3723	01110	CALL	EVAL
43AD	E5	01120	PUSH	HL
43AE	CD7F0A	01130	CALL	SHAPE
43B1	EB	01140	EX	DE,HL
43B2	E1	01150	POP	HL
43B3	DDE1	01160	POP	IX
43B5	CD5043	01170	CALL	DIXIE
43B8	DD360000	01180	LD	(IX),0
43BC	DD23	01190	INC	IX
43BE	F1	01200	POP	AF
43BF	B7	01210	OR	A
43C0	C0	01220	RET	NZ
43C1	DD224E43	01230	LD	(TNEXT), IX
43C5	C9	01240	RET	
43C6	D1	01500	USR POP	DE
43C7	23	01510	INC	HL
43C8	CD5B43	01520	CALL	SCAN
43CB	B7	01530	OR	A
43CC	CA9719	01540	JP	Z, SNERR
43CF	DD5E00	01550	LD	E, (IX)
43D2	DD5601	01560	LD	D, (IX+1)
43D5	D5	01570	PUSH	DE
43D6	FDE1	01580	POP	IY
43D8	0600	01590	LD	B,0
43DA	09	01600	ADD	HL,BC
43DB	2B	01605	DEC	HL
43DC	D7	01610	RST	16
43DD	CF	01620	RST	8
43DE	2B	01630	DEFB	40
43DF	DD21A042	01640	LD	IX, STORE
43E3	DDE5	01650	PUSH	IX
43E5	CD0D26	01660	LP5 CALL	VARPT
43E8	CD5043	01670	CALL	DIXIE
43EB	2B	01680	DEC	HL
43EC	D7	01690	RST	16
43ED	FE2C	01700	CP	44
43EF	2003	01710	JR	NZ, NXT3

43F1	D7	01720	RST	16	
43F2	18F1	01730	JR	LP5	
43F4	CF	01740	NXT3 RST	8	
43F5	29	01750	DEFB	41	
43F6	DDE1	01760	POP	IX	
43F8	E5	01770	PUSH	HL	
43F9	21900B	01780	LD	HL, RETN	
43FC	E5	01790	PUSH	HL	
43FD	FDE9	01800	JP	(IY)	
0C77		02000	DADD	EQU	0C77H
0C70		02010	DSUB	EQU	0C70H
0DA1		02020	DMULT	EQU	0DA1H
0DE5		02030	DDIV	EQU	0DE5H
09D3		02040	MOVE	EQU	09D3H
09FC		02050	SWAP	EQU	09FCH
0A4F		02060	CPACC	EQU	0A4FH
40AF		02070	NTF	EQU	40AFH
411D		02080	ACC	EQU	411DH
4127		02090	AUX	EQU	4127H
42BF		02300	FLAG	EQU	42BFH
42C0		02310	VAR	EQU	42C0H
42C8		02320	CUM	EQU	42C8H
42D0		02330	TOT	EQU	42D0H
42D8		02340	NUM	EQU	42D8H
42E0		02345	SPARE	EQU	42E0H
43FF	0000	02350	ONE	DEFW	0
4401	0000	02360	DEFW	0	
4403	0000	02370	DEFW	0	
4405	0081	02380	DEFW	8100H	
4407	C468	02390	PI2 DEFW	68C4H	
4409	21A2	02400	DEFW	0A221H	
440B	DA0F	02410	DEFW	0FDAH	
440D	4981	02420	DEFW	8149H	
440F	CAE9	02430	PI180 DEFW	0E9CAH	
4411	9412	02440	DEFW	1294H	
4413	35FA	02450	DEFW	0FA35H	
4415	0E7B	02460	DEFW	7B0EH	
4417	7BCF	02462	LOG2 DEFW	0CF7BH	
4419	D1F7	02464	DEFW	0F7D1H	
441B	1772	02466	DEFW	7217H	
441D	3180	02468	DEFW	8031H	
441F	21AF40	02500	SETUP1 LD	HL, NTF	
4422	3608	02510	LD	(HL), 8	
4424	3EFF	02520	LD	A, 255	
4426	32BF42	02530	LD	(FLAG), A	
4429	DD5E00	02540	LD	E, (IX)	
442C	DD5601	02550	LD	D, (IX+1)	
442F	CD8744	02560	CALL	DOIT1	
4432	1831	02570	JR	OUTVAR	
4434	CD6A44	02575	NUMUP CALL	NDVCUM	
4437	CDFC09	02580	TOTUP CALL	SWAP	
443A	11D042	02582	LD	DE, TOT	
443D	CD8744	02584	CALL	DOIT1	
4440	CDC844	02590	CALL	SWITCH	
4443	181B	02600	JR	OUTTOT	

4445	CD1F44	02610	EXPCOS	CALL	SETUP1
4448	CDEB44	02620	SETUP3	CALL	NUMONE
444B	AF	02630		XOR	A
444C	32DF42	02640		LD	(NUM+7),A
444F	CD7F44	02645		CALL	ONEACC
4452	1809	02650		JR	SETUP4
4454	CD6544	02655	OUTSET	CALL	OUTVAR
4457	CDEB44	02660	SETUP2	CALL	NUMONE
445A	CD7544	02670		CALL	VARACC
445D	CD6D44	02680	SETUP4	CALL	OUTCUM
4460	21D042	02690	OUTTOT	LD	HL, TOT
4463	180B	02700		JR	DOIT3
4465	21C042	02710	OUTVAR	LD	HL, VAR
4468	1806	02720		JR	DOIT3
446A	CDC244	02730	NDVCUM	CALL	NUMDIV
446D	21C842	02740	OUTCUM	LD	HL, CUM
4470	111D41	02750	DOIT3	LD	DE, ACC
4473	1815	02760		JR	MOVIT
4475	11C042	02770	VARACC	LD	DE, VAR
4478	180D	02780		JR	DOIT1
447A	11C842	02790	CUMACC	LD	DE, CUM
447D	1808	02800		JR	DOIT1
447F	11FF43	02810	ONEACC	LD	DE, ONE
4482	1803	02820		JR	DOIT1
4484	110744	02830	PIACC	LD	DE, PI2
4487	211D41	02840	DOIT1	LD	HL, ACC
448A	C3D309	02850	MOVIT	JP	MOVE
448D	11C042	02860	VARAUX	LD	DE, VAR
4490	180B	02870		JR	DOIT2
4492	CD7A44	02880	CUMTOT	CALL	CUMACC
4495	11D042	02890	TOTAUX	LD	DE, TOT
4498	1803	02900		JR	DOIT2
449A	11FF43	02910	ONEAUX	LD	DE, ONE
449D	212741	02920	DOIT2	LD	HL, AUX
44A0	18E8	02930		JR	MOVIT
44A2	11D842	02940	NUMAUX	LD	DE, NUM
44A5	18F6	02950		JR	DOIT2
44A7	CD1F44	02960	CONV	CALL	SETUP1
44AA	110F44	02970	PIAUX	LD	DE, PI180
44AD	18EE	02980		JR	DOIT2
44AF	CDB944	03000	CALCX	CALL	CALCY
44B2	180B	03010		JR	VARMUL
44B4	CDE344	03020	UPACC	CALL	UPNUM
44B7	18C1	03030		JR	CUMACC
44B9	CDB444	03035	CALCY	CALL	UPACC
44BC	CDBD44	03040	VARMUL	CALL	VARAUX
44BF	C3A10D	03050		JP	DMULT
44C2	CDA244	03060	NUMDIV	CALL	NUMAUX
44C5	C3E50D	03070		JP	DDIV
44C8	3ABF42	03080	SWITCH	LD	A, (FLAG)
44CB	2F	03090		CPL	
44CC	32BF42	03100		LD	(FLAG), A
44CF	B7	03110	FLIP	OR	A
44D0	2805	03120		JR	Z, SUBRET
44D2	180C	03130		JR	ADDRET

44D4	CD9A44	03140	SUBONE	CALL	ONEAUX
44D7	C3700C	03150	SUBRET	JP	DSUB
44DA	CD7F44	03160	NUMADD	CALL	ONEACC
44DD	CDA244	03170	ADDNUM	CALL	NUMAUX
44E0	C3770C	03180	ADDRET	JP	DADD
44E3	CDDA44	03181	UPNUM	CALL	NUMADD
44E6	111D41	03182		LD	DE, ACC
44E9	1803	03184		JR	NUMB
44EB	11FF43	03190	NUMONE	LD	DE, ONE
44EE	21D842	03200	NUMB	LD	HL, NUM
44F1	1897	03210		JR	MOVIT
44F3	CD4544	03500	DEXP	CALL	EXPCOS
44F6	3E7F	03503		LD	A, 127
44FB	CDB944	03505		LD	(FLAG), A
44FB	CDB944	03510	LPX	CALL	CALCY
44FE	CD3444	03550		CALL	NUMUP
4501	3ACF42	03560		LD	A, (CUM+7)
4504	B7	03570		OR	A
4505	C8	03580		RET	Z
4506	18F3	03590		JR	LPX
4508	CD5745	03600	DSQR	CALL	DLOG
450B	212441	03610		LD	HL, ACC+7
450E	35	03620		DEC	(HL)
450F	CD6544	03625		CALL	OUTVAR
4512	CD4844	03630		CALL	SETUP3
4515	18E4	03640		JR	LPX
4517	CD1F44	03700	DSIN	CALL	SETUP1
451A	CD5744	03710		CALL	SETUP2
451D	1803	03720		JR	CALC1
451F	CD4544	03800	DCOS	CALL	EXPCOS
4522	CDAF44	03900	CALC1	CALL	CALCX
4525	CD6A44	03930		CALL	NDVCUM
4528	CDB444	03935		CALL	UPACC
452B	CD3444	03940		CALL	NUMUP
452E	3ACF42	03950		LD	A, (CUM+7)
4531	B7	03960		OR	A
4532	C8	03970		RET	Z
4533	18ED	03980		JR	CALC1
4535	CD1F44	04090	SETUP5	CALL	SETUP1
4538	3A2341	04100		LD	A, (ACC+6)
453B	FEB0	04110		CP	80H
453D	D8	04120		RET	C
453E	D1	04130		POP	DE
453F	C39719	04140		JP	SNERR
4542	CD1F45	04200	DTAN	CALL	DCOS
4545	21E042	04210		LD	HL, SPARE
4548	CD7044	04220		CALL	DOIT3
454B	CD1745	04230		CALL	DSIN
454E	11E042	04240		LD	DE, SPARE
4551	CD9D44	04250		CALL	DOIT2
4554	C3E50D	04260		JP	DDIV
4557	CD3545	04600	DLOG	CALL	SETUP5
455A	212441	04602		LD	HL, ACC+7
455D	7E	04604		LD	A, (HL)
455E	D682	04606		SUB	130

4560	32E042	04608	LD	(SPARE),A
4563	3682	04610	LD	(HL),130
4565	CD9A44	04620	CALL	ONEAUX
4568	CD770C	04630	CALL	DADD
456B	CD6544	04640	CALL	OUTVAR
456E	CDD444	04650	CALL	SUBONE
4571	CDD444	04660	CALL	SUBONE
4574	CD8D44	04670	CALL	VARAUX
4577	CDE50D	04680	CALL	DDIV
457A	CD5444	04690	CALL	OUTSET
457D	3E7F	04692	LD	A,127
457F	32BF42	04694	LD	(FLAG),A
4582	CDE845	04700	CALL	CALC2
4585	CD9544	04710	CALL	TOTAUX
4588	CD770C	04720	CALL	DADD
458B	21E042	04722	LD	HL,SPARE
458E	7E	04724	LD	A,(HL)
458F	B7	04726	OR	A
4590	C8	04728	RET	Z
4591	F29845	04730	JP	P,POS
4594	3600	04732	LD	(HL),0
4596	ED44	04734	NEG	
4598	47	04736	LD	B,A
4599	D9	04737	EXX	
459A	111744	04738	LD	DE,LOG2
459D	CD9D44	04740	CALL	DOIT2
45A0	3AE042	04742	LD	A,(SPARE)
45A3	CDCF44	04744	CALL	FLIP
45A6	D9	04745	EXX	
45A7	10F0	04746	DJNZ	LPL
45A9	C9	04748	RET	
45AA	CD1F44	04800	CALL	SETUP1
45AD	CD9A44	04810	CALL	ONEAUX
45B0	CD4FOA	04820	CALL	CPACC
45B3	FE01	04830	CP	1
45B5	2B12	04840	JR	Z,OVER
45B7	B7	04850	OR	A
45B8	2B24	04860	JR	Z,UNIT
45BA	212D41	04870	LD	HL,AUX+6
45BD	3680	04880	LD	(HL),12B
45BF	CD4FOA	04890	CALL	CPACC
45C2	FE01	04900	CP	1
45C4	2B1F	04910	JR	Z,UNDER
45C6	B7	04920	OR	A
45C7	2B15	04930	JR	Z,UNIT
45C9	CDFC09	04940	CALL	SWAP
45CC	CD7F44	04950	CALL	ONEACC
45CF	CDE50D	04960	CALL	DDIV
45D2	CDE545	04970	CALL	UNDER
45D5	CDFC09	04980	CALL	SWAP
45D8	CD8444	04990	CALL	PIACC
45DB	C3700C	05000	JP	DSUB
45DE	CD8444	05010	CALL	PIACC
45E1	2B	05020	DEC	HL

45E2	35	05030	DEC	(HL)
45E3	1822	05040	JR	MULRET
45E5	CD5444	05050	UNDER	CALL
45E8	CDE344	05100	CALLC2	CALL
45EB	CDAF44	05110	CALL	UPNUM
45EE	CD6D44	05120	CALL	CALCX
45F1	CDC244	05130	CALL	OUTCUM
45F4	CD3744	05140	CALL	NUMDIV
45F7	3ACF42	05150	LD	TOTUP
45FA	B7	05160	OR	A,(CUM+7)
45FB	C8	05170	RET	A
45FC	18EA	05180	JR	Z
45FE	CDA744	05180	JR	CALC2
4601	C3E50D	05300	DD	CALL
4604	CDA744	05310	JP	CONV
4607	C3A10D	05400	DR	DDIV
460A	CD8444	05410	MULRET	CALL
460D	2B	05500	DPI	CONV
460E	34	05510	DEC	DMULT
460F	C9	05520	INC	CALL
4610	00	05530	RET	PIACC
4611	0000	05540	DEFB	HL
4613	00	05600	FIN	(HL)
40A4		05610	FIN1	0
40A4		05700	ORG	0
40A4	1146	05710	DEFW	40A4H
40F9		05720	ORG	FIN
40F9	1346	05730	DEFW	40F9H
40FB	1346	05740	DEFW	FIN1
40FD	1346	05750	DEFW	FIN1
42E9		05800	ORG	FIN1
42E9	00DD	05810	DEFW	START
42EB	0845	05820	DEFW	ODD00H
42ED	00DF	05830	DEFW	DSQR
42EF	5745	05840	DEFW	DEFW
42F1	00E0	05850	DEFW	ODFO0H
42F3	F344	05860	DEFW	DLOG
42F5	00E1	05870	DEFW	OE000H
42F7	1F45	05880	DEFW	DEXP
42F9	00E2	05890	DEFW	OE100H
42FB	1745	05900	DEFW	DCDS
42FD	00E4	05910	DEFW	OE200H
42FF	AA45	05920	DEFW	DSIN
4301	00E3	05930	DEFW	DEFW
4303	4245	05940	DEFW	OE400H
4305	0044	05950	DEFW	DATN
4307	FE45	05960	DEFW	OE300H
4309	0052	05970	DEFW	DTAN
430B	0446	05980	DEFW	4400H
430D	0050	05990	DEFW	DD
430F	0A46	06000	NEW	5200H
4311	00	06010	DEFW	DR
003C		06200	DEFB	5000H
434E	1243		DEFW	DPI
06CC			DEFB	0
			DEFW	60
			END	NEW
				06CCH

***** NEXT MONTH'S ISSUE *****

Next month's issue will contain at least the following programs plus the usual features and articles.

** VDU SAVE L2/4K **

This BASIC program creates a machine language program that allows the screen display to be saved into memory and then be recalled at a later time. Two new commands are added to non-disk BASIC, LSET to store the image and RSET to recall an image.

** FLASHING MESSAGE L2/16K m.1. **

A machine language program which displays a flashing message at the top of the screen independent of the BASIC program you are running. The rate of flash can be adjusted and the message can be turned on or off from BASIC.

** MIND READER L2/4K **

Can this program read your mind? Is this artificial intelligence?...(or is it just computer trickery). Next month you get the chance to find out, if you can!

** INCOME TAX L2/4K **

Before you know where you are, it will be that time of year again, so be ready for it with this program which will calculate the income tax payable for a person who would normally complete the Australian "S" Income Tax return.

** LONGVARS L2/48K m.1. **

This is a machine language pre-processor for BASIC programs. It will allow you to use long variable names in a BASIC program, (in the development stage) so that names like AARDVARK and AARDWOLF will be distinguished. Also, variable names that contain reserved words may be used, e.g. STIFF which contains IF.

- 0000000000 -

***** MICROBUGS *****

In which we correct those errors which seem to creep in, no matter how careful we are.

BASIC + LABELS Vol. 3 Issue 1, December 1981.

Oops, in the voluminous documentation which accompanied this machine language program we omitted to mention the entry point, thus leaving you with 2000 odd bytes of unexecutable code. The entry point is 7E42H(32386 Dec).

- 0000000000 -

APPLICATION FOR PUBLICATION OF A PROGRAM IN MICRO-80

Date

To MICRO-80 SOFTWARE DEPT. P O BOX 145 MORPHETTVALE SA 5162
Please consider the enclosed program for ...

Tick where appropriate

(i) Publication in MICRO-80

(ii) Publication on disk or cassette only

(iii) Both

Name

Address

Postcode

*** CHECK LIST ***

Please ensure that the cassette or disk is clearly marked with your name and address, program name(s), Memory size, Level I, II, System 1 or 2, Edtasm, System, etc. The use of REM statements with your name and address is suggested, in case the program becomes separated from the accompanying literature.

Ensure that you supply adequate instructions, notes on what the program does and how it does it, etc.

For system tapes, the start, end, and entry points, etc.

The changes or improvements that you think may improve it.

Please package securely - padabags are suggested - and enclose stamps or postage if you want your cassette or disk returned.

***** CASSETTE/DISK EDITION INDEX *****

The cassette edition of MICRO-80 contains all the software listed each month, on cassette. All cassette subscribers need do is CLOAD and RUN the programs. Level II programs are recorded on side 1 of the cassette. Level I programs are recorded on side 2. Level I programs are not compatible with the System 80. All programs are recorded twice in succession. Note, System 80 computers have had different tape-counters fitted at different times. The approximate start positions shown are correct for the very early System 80 without the volume control or level meter. They are probably incorrect for later machines. The rates for a cassette subscription are printed on the inside front cover of each issue of the magazine.

The disk edition contains all those programs which can be executed from disk, including Level I programs. Level I disk programs are saved in the NEWDOS format. Users require the Level I/CMD utility supplied with NEWDOS + or NEWDOS 80 version 1.0 to run them.

SIDE ONE	TYPE	I.D.	DISK FILESPEC	APPROX. START POSITION		
				CTR-41	CTR-80	SYSTEM-80
TESTSTR (Demonstration)	L2/4K	T	TESTSTR/BAS	15	10	10
"	"	"	"	22	15	16
MEASUREMENTS	L2/16K	M	MEASURE/BAS	30	21	22
"	"	"	"	105	71	75
SUPER HANGMAN	L2/16K	H	HANGMAN/BAS	170	116	120
"	"	"	"	255	171	180
DUPLEX	SYSTEM	DUPLEX	DUPLEX/CMD	325	221	230
"	"	"	"	335	226	237
SIDE TWO						
DATA BASE MANAGEMENT SYSTEM	48K/DISK	D	DBMS/BAS	15	10	10
"	"	"	"	97	65	68
FILES	L2/48K	F	FILES/BAS	170	114	120
"	"	"	"	210	141	148
DUPLEX	EDTASM	DUPLEX	DUPLEX/EDT	250	167	175
"	"	"	"	295	198	208

TO:
 MICRO-80, P.O. BOX 213, GOODWOOD,
 SOUTH AUSTRALIA, 5034.
 Please RUSH to me the items shown below:

\$ enclosed Date

..... 12 month subscription to MICRO-80

..... 12 month subscription to MICRO-80, plus the cassette edition

The latest issue of MICRO-80 (see inside front cover for prices)
 The MICRO-80 PRODUCTS listed below:

DESCRIPTION	PRICE

TOTAL ENCLOSED WITH ORDER

Cheque Bankcard Money Order

Bankcard Account Number

--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

Signature Exp. End

NAME

ADDRESS

Postcode



Tandy's TRS-80 Color Computer Adds Color And Sound To Personal Computing!

NEW to Australia! Tandy Electronics introduces the TRS-80 Color Computer.

Set to add a new and exciting dimension to the world of personal computing, the addition of colour and sound will make programmes come alive as you and your family enter a new era of the computer revolution.

Tandy's TRS-80 Color Computer is complemented by an extensive range of ready-to-run software; business, personal management, educational, and entertainment programmes that will involve every member of the family.



NOW available for immediate delivery, the TRS-80 4K Color Computer (expandable up to 32K) starts from a low \$599* . 269-3001

Tandy
ELECTRONICS

"THE BIGGEST NAME IN LITTLE COMPUTERS"

Available through Tandy Computer Centres, Computer Departments, and participating Tandy Dealers. *Monitor (not included)

MICRO-80