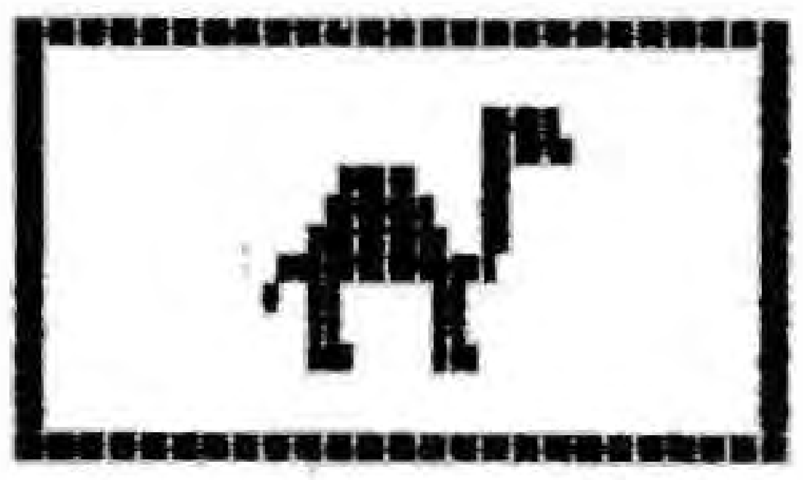
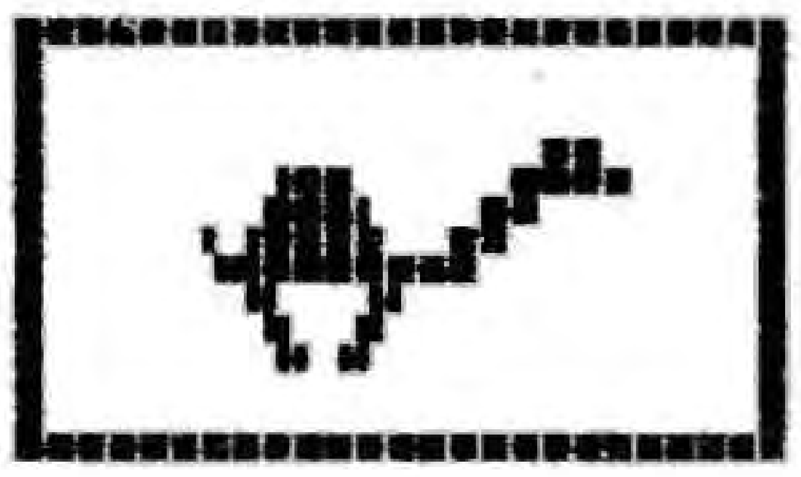
MICRO-80

P.O. BOX 213, GOODWOOD, S.A. 5034. AUSTRALIA. TELEPHONE (08) 211 7244. PRICE: AUS. $2.50, N.Z. $4.00, U.K. £1.50
Registered by Australia Post — Publication No. SBQ2207

Vol. 4, Issue 2, September 1983

DESERT CHASE for Level II


YOUR CAMEL
LIKES THIS PACE


YOUR CAMEL IS
BURNING ACROSS
THE DESERT SANDS
WARNING —
YOU NEED A DRINK

YOU HAVE ARRIVED
AT A WATERHOLE...
YOUR CAMEL EATS
HAPPILY WHILE YOU
REFILL YOUR
WATER-BAG

• TRS-80 • SYSTEM 80 • VIDEO GENIE
• PMC-80 • HITACHI PEACH
• TRS-80 COLOUR COMPUTER

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

EDITOR:	RYSZARD WIWATOWSKI
ASSOCIATE EDITORS:	
SOFTWARE :	CHARLIE BARTLETT
HARDWARE :	EDWIN PAAY

MICRO-80 is an international magazine devoted to the Tandy TRS-80 Model I, Model III and Colour microcomputers, the Dick Smith System 80/Video Genie and the Hitachi Peach. It is available at the following prices:

	12 MONTH SUB.	SINGLE COPY
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

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

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)	MAGAZINE	CASS + MAG	DISK + MAG
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/Video Genie or Peach and its peripherals. MICRO-80 is in no way connected with any of the Tandy, Dick Smith or Hitachi 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 microcomputer to earn some extra income is included in every issue.

** CONTENT **

Each month we publish at least one applications program in BASIC for each of the microcomputers we support. We also publish Utility programs in BASIC and Machine Language. We publish articles on hardware modifications, constructional articles for useful peripherals, articles on programming techniques both in Assembly Language and BASIC, new product reviews for both hardware and software and we print letters to the Editor.

** COPYRIGHT **

All the material published in this magazine is under copyright. This 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 *****

MICRO-80 is Registered by Australia Post - Publication No. SBQ2207

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, 379 South Road, MILE END, S.A., 5031
Published in Australia by:	MICRO-80 433 Morphett Street, Adelaide, SA., 5000

***** EDITORIAL *****

With the announcement of a new range of computers, Tandy Australia offers something to interest everyone from the MC-10 Micro Colour Computer for $199.95 (available in November), through the new Model 4 to the Models 12 and 16. There's even a portable computer for the travelling businesman.

The word 'portable' takes on a new meaning with the Model 100 which is about the size of its own manual but lighter. It has a LCD display of 8 lines of 40 characters (with graphics capability) and is battery powered with a non-volatile memory. The basic hardware consists of a good keyboard, 8K of RAM (expandable to 24K), a clock with calendar, and interfaces to accommodate a Centronics printer, a RS-232-C port and a cassette recorder for data storage. The 32K of ROM provides a BASIC interpreter, a word processor, an address and appointment facility and communications software. Once you become familiar with these, I image that you can take and use the Model 100 literally anywhere.

The new Model 4 will be available in three configurations - 16K cassette (for $1,799), 64K single drive (for $2,799) and 64K two drive (for $3,299). Model 3 owners can upgrade their computers with a kit that includes a new keyboard, a new CPU board with 64K memory and the TRSDOS 6.0 operating system with Disk BASIC and sells for $1,299. CP/M Plus for the Model 4 is expected within two to three months.

Meanwhile, in the U.S. Tandy has released a new, smart-looking 64K Colour Computer (featuring a good quality keyboard) and the OS-9 real-time, multi-tasking, multi-user operating system providing access to a broad range of existing applications software. The Multi-Pak Interface (priced around $US180) is an accessory allowing the Colour Computer user to select one of four ROM packs at the flick of a switch or under software control. Undoubtedly these too will be available in Australia in the near future.

- 0000000000 -

***** PEEKing (UK) - by Tony Edwards *****

You will remember the universal programming language NOS-BASICODE reported in an earlier issue of MICRO-80. Rather than die, this language is still developing and the latest development is NOS-BASICODE-2, which includes the earlier version. These programming languages have their origin in the Dutch Broadcasting Corporation's program "Hobbyscoop" (pronounced Hobby Scope). Programs on computing, including the transmission of programs in BASICODE are output regularly by Radio Hilversum 2 and 4 on the medium and VHF wave bands for reception in Holland, but these are easily received in other European Countries.

For our readers in other parts of the world short-wave transmissions were tried but were not successful. However, the programs are now transmitted to radio stations world wide for re-transmission locally. The program to look for is "Radio Activity", a 15 minute English language program which is re-transmitted for local reception in Australia, USA, Canada, and parts of Africa and Asia, as well as Europe. The local stations broadcast at suitable local times but the internal Dutch program is transmitted at 1710 GMT on Sundays in the summer, and at 1810 on Sundays in the winter. It will be found at 747kHz (401 metres). Good listening! The main program is in Dutch but it includes short explanations in English.

- 0000000000 -

***** INPUT/OUPUT *****

From: John Veldthius - Taranaki, N.Z.

Let me first say what a fantastic magazine you have. Now for my question. I have bought the adventure game called Asylum and have been going nutty for the last two months. I am stuck in the first maze and cannot find any way out. I have read the hints in Vol. 3, Issue 11 (October 1982) but these have been of no help at all. I have mapped the maze so far but still can't find anything else.

Please help as I may end up in the "Institute" at this rate.

From: Steven Bauer - Upwey, Vic.

These are questions which, hopefully, you will answer or, if not, maybe one of your other readers will be able to help.

Asylum - how do I get out of the first maze; please explain each and every step.

To Brian L. Hill - Where did you buy the components? (for the 32K mod). Dick Smith's price is over $23.

From: Nick Lambropoulos - Mildura, Victoria.

It's getting a bit monotonous, I know, but I'm hoping, before everyone packs up and forgets about Asylum (?), which is usually widely talked about nowadays. I CRY OUT one final time for someone to tell me whereabouts the matches could be found in the Asylum, if it's not too much trouble. I'm also having trouble figuring out what the hint means in the Guru's room and what is in the other Dark Room. I would appreciate any help given. Keep up the great work, Micro-80.

From: Jeremy Terhoeve - Townsville, Qld.

Please can anyone tell me how to get past the gorilla in the second maze of Asylum? I cannot fully understand the hint the program gives here. Do you need a special object?

(To answer these questions, we need to refer to a professional by the name of Ron Hack who was glad to help by providing specific answers. Thanks very much, Ron. -Ed.)

From: Ron Hack - Reynella, S.A.

To assist those readers still seeking clues for the "Asylum" adventure and to refresh my own memory, I decided to play the adventure again and answer the questions asked as I proceeded. Here goes -

GENERAL HINTS:

As you are aware, the adventure must be completed within a certain time. If you wish to take a break during the game (for a cup of coffee or to write some clues for a magazine) call up the vocabulary as this stops the clock.

To John Veldthius, Taranaki, N.Z.

On a large blank piece of paper draw light pencil lines across and down the page equal spaces apart, making a grid. As you proceed to move down the passages draw heavy lines marking the walls as if you were looking down from above. Remember you will hyperspace in certain places and dropping things along the way will help you to sort out where this occurs.

To Steven Bauer, Upwey, Vic.

It is impossible to give step by step instructions on getting out of the 1st maze (it would take up too much magazine space and possibly spoil the adventure for others).

Hints:
- You will need to find a key
- Murderers can read
- There are eight different corridors off the revolving doors and not four as I thought for a long time.
- Having mapped all the maze you should find it forms a rectangle.

To Nick Lambropoulos, Mildura, Vic.

The hint in the Guru's room is in Latin (so I've been told) and translated roughly means - In the dark, horses look like cows (and, as we know, donkeys look like horses). This tends to mean something once you meet the inmate outside the room full of water. Having met this inmate I suggest you try some trading.

The matches are in the 2nd maze but you will have to solve the mystery of the 20 room corridor before you find them. Refer to earlier clues (MICRO-80 Oct. '82).

As to the other dark room and its contents - if you are curious try lighting a match to find out.

To Jeremy Terhove, Townsville, Qld.

You do need an object to help you pass the gorilla in the 2nd maze and you'll find it in that maze. You also need an object you are already carrying assuming, of course, you have not dropped something since entering 2nd maze. The hint given at this stage of the adventure is a good clue as to what object may help you.

To Grant Barnes, Moe, Vic.

You are wasting your time looking for the Professor's Office unless you are up to this stage of the adventure. For what it's worth, the office is in the final small maze and when you reach that stage you will find it easily enough.

Well, that's it for now. I'll just get in this strange contraption and.......WHEEEE - I wonder where I'll land!!

- 0000000000 -

DEPARTMENTS

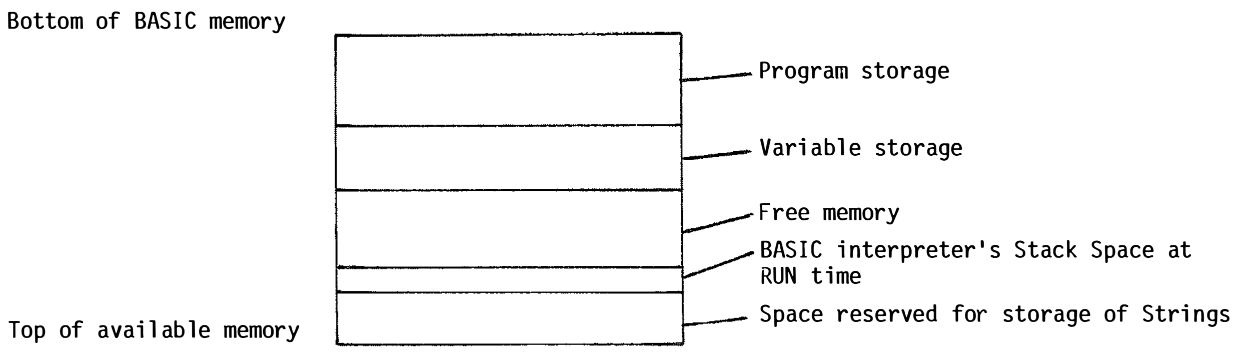
***** KALEIDOSCOPE *****

In this month's issue we publish for the first time a program for the Colour Computer (Othello) submitted by one of our readers and we look forward to bringing you more programs from our readers that show how they use their computers and how to improve your programming skills and your knowledge of the Colour Computer. Towers of Hanoi - an interesting puzzle for you to try on your computer - demonstrates the use of recursive programming in BASIC and is worth looking at more closely. For those of you who cannot find the time to type in the programs from the magazine listings, and would prefer an easier method, we have good news - the Cassette Edition of the magazine, commencing with this issue, will include all the Colour Computer programs contained in that issue! The cost of the magazine, plus cassette subscription with be $65 for 12 issues and single cassettes will be available for $4.00 each.

Most of the Level 2 programs that we have converted are intended to run on a 16K Colour Computer with Extended Colour BASIC, but we seem to have slipped up with one of these quite unintentionally. Here at MICRO-80 our Colour Computer consists of a 32K Extended Colour BASIC machine with one disk drive. The Level 2 programs are transferred, edited to remove the reserved words not used in Colour BASIC and to accommodate the smaller 32 x 16 display, and then tested on this machine. In most cases, what was a 16K Level 2 program works fine on a 16K Colour Computer, but in the case of "Sink the Enemy Navy", the memory required to run the program crept just over the 16K. How can you get a program to use less memory? Read on to find out.

BASIC PROGRAM MEMORY USAGE

To run a BASIC program, four well-defined and distinct uses are made of the available memory in your machine. These are illustrated in the following diagram:



BASIC MEMORY USAGE

When you type in your BASIC program, the lines are tokenized to conserve memory and stored in the program storage area. However, if the program is to run successfully (assuming there are no programming or syntax errors) sufficient memory must be available for storing program variables, String storage and the BASIC stack, otherwise the program will stop during execution with an Out of Memory or Out of String Space error (the BASIC interpreter constantly checks the amount of free memory and, if it falls below a certain lower limit, terminates the program with an error).

REDUCING MEMORY REQUIREMENT

When a program halts with an OM error, the answer is not necessarily to rush out and buy more memory (an expensive solution but one almost guaranteed to work). You can overcome the problem by making more efficient use of your available memory by a combination of the following methods. Naturally, there are compromises in some of these and a few violate what is known as "good programming practice" but I think that under these circumstances the end justifies the means.

1. Program Storage

Comments within a program are ignored by BASIC and serve only as documentation for the programmer. If you run out of memory, start by removing all comments from the program (but keep a backup copy with the comments for future reference). However, be careful that you do not introduce other errors by doing this. For example, consider the following program segment:

```
20 REM CHECK KEYBOARD
30 A$=INKEY$:....etc.
.
.
100 GOSUB 20
```

Deleting the remark line 20 will produce an error at run time since line 100 references a non-existent line number. Line 20 can be deleted but line 100 must also be changed to GOSUB 30 to produce the desired effect.

Each BASIC line incurs a 5 byte overhead for the interpreter's use. By placing several statements on one line you can save a good deal of memory in the program storage area. For example, the following portion of the program "Sink the Enemy Navy" could be rewritten to use much less memory:

```
 30 DIM C(14,9)
 40 DIM B1(5)
 50 DIM B2(5)
 60 DIM C1(5)
 70 DIM C2(5)
 80 DIM D1(5)
 90 DIM D1(5)
100 DIM S1(5)
110 DIM (S2(5)
```

can be replaced by:

```
30 DIM C(14,9), B1(5), B2(5), C1(5), C2(5), D1(5), D2(5), S1(5), S2(5)
```

The total memory saved amounts to 40 bytes, not to mention the slightly faster execution time!

2. String Storage

The amount of string space used and the amount required by a program are not necessarily the same. However, there is good reason for allocating a generous amount of string space. During program execution the interpreter makes use of string space somewhat extravagantly. To illustrate, suppose A$ consists of the first 25 letters of the alphabet and you want to append the letter "Z". You can do this with the statement A$=A$+"Z". What actually happens in the String Storage section is that the first string of 25 characters is discarded as garbage and a new string of 26 characters is created and stored. If the amount of the string space is exhausted, the interpreter begins a "garbage collection" routine. Frequent use of this routine slows down program execution speed. The moral of the story is that if you can afford to be extravagant then allow a generous amount of string space in your CLEAR statement, but if memory is scarce, use only what is required.

3. Variable Storage

Each time a new variable is used within a program additional memory is used in the variable storage area. When writing a large program you can be reasonably certain that it will use most of your available memory. It would be wise to consider your total variable requirements (both in number and in use) at the outset and try to use as few as possible to conserve memory and improve the execution speed of the program. As a general rule, try to re-use existing variables whenever possible because each new variable used within a program consumes some memory for the duration of the program, even if it is used only once. It is much easier to consider these needs in the initial stages of program development than it is to make extensive changes to variables in an almost complete program.

There are occasions where the creation of a new variable can actually be used to save memory. To illustrate, suppose your program uses the number 20000 frequently, say, for example twenty times in different parts of the program. You can save a substantial amount of memory (about 80 bytes) by assigning V=20000 at the beginning of the program and using the variable V in place of the number 20000 elsewhere in the program. Not only does this save memory, but as a bonus, it increases the speed of the program and saves some typing as well.

IN SUMMARY

The place to start when you're trying to save some memory is within the program storage area. More often than not, you can free enough memory here to use the program effectively without having to resort to more drastic measures. If you must economize in variable usage and string space allocation, then perhaps the option of increasing your amount of physical memory should be considered. There are other ways to conserve memory such as abbreviating instructions and messages, or deleting parts of the program that you can do without, but these are more a matter of personal taste rather than techniques of programming. So if you've been frustrated by some large program that just wouldn't fit your 16K Colour Computer, try trimming some of the fat!

- 0000000000 -

***** PEACH BOWL *****

Many who are new to BASIC find programming in that language difficult and the idea of programming in assembly language (or machine language) seems totally out of the question. This month we present "Register Display" which is a reasonably short machine language utility that, apart from being useful, may serve to teach you a little about this low-level programming language. When you wish to learn how to program in machine language the place to start is with a good book that explains the fundamentals and the instruction set of the processor you're using (in this case, the Motorola 6809). A good example is the book "Programming the 6809" by Rodnay Zaks which tends to be less formidable than Motorola's own publications. The Peach provides an adequate machine code monitor that will allow you to develop and test short m.l. programs that you have assembled by hand.

For those of you who like puzzles, we have a version of "Towers of Hanoi" for the Peach that should keep you amused for hours. The original puzzle would take innumerable lifetimes to solve, but this version is somewhat abbreviated. And good news for those of you who cannot find the time to type in those long magazine listings - our cassette edition now includes the Peach programs published in that issue of the magazine, commencing with this issue. The prices for cassettes will be the same as shown inside the front cover - $4.00 for a single cassette and $65.00 for a subscription that includes magazine plus cassette.

SECTOR EDITOR CORRECTION

If you had any problems with this program after making the amendments published last issue, then perhaps this correction may solve that problem:

Change

```
    6Ø13Ø DATA 36,3Ø,1Ø,C6,ØF,34,...
```

to

```
    6Ø13Ø DATA 36,3Ø,1Ø,C6,1Ø,34,...
```

Geof Drury, who developed the amendments to Sector Editor, also passed on this correction.

The discussion on conserving memory in Kaleidoscope will also be of interest to new Peach users. (The Peach too uses a Microsoft BASIC interpreter). In the case of the Peach, it may be added that although the use of long variable names does improve the clarity of a program, more memory is used to store the longer name and, in the case of very long programs, execution speed increases.

- 0000000000 -

***** GROUP ONE *****

While 'Desert Chase' and 'Ordering Priorities' are relatively straightforward Level 2 programs and should run in disk systems without changes, the machine language utility 'Formation' is definitely for Level 2 systems only as presented. For disk users, the changes required to use this utility with disk BASIC involve relocating the program (not very difficult if you have an Editor/Assembler), modifying the initialiser and using a different reserved word (such as NAME which doesn't appear to be used by some disk BASICs) and rewriting the I/O routines to make use of disk instead of cassette. We would be pleased to publish any adaptations developed by our disk system readers.

If you are a novice computer enthusiast, or even perhaps a more seasoned user, you may find the discussion on conserving memory in this month's Kaleidoscope of particular interest. All of the microcomputers that we support use a version of Microsoft BASIC and the information presented there is equally applicable to the Z80-based machines.

THE CASE OF THE LOWER CASE

At some stage in our relationships with our computers, many of us will take a step further inside the machine and begin to dabble with a hardware modification of some sort. More often than not, the apparent simplest of modifications will lead to a small hiccup that gradually grows to mammoth proportions the longer we struggle to overcome the difficulty. Mr. Wilson, one of our readers relates one such experience:

> " I have installed a lower case modification as described in "THE CUSTOM TRS-80 & OTHER MYSTERIES" which was advertised and purchased through your magazine by a friend of mine. The mod was successful, because running the sample program gives me lower case letters.
>
> Also running the short program as described in MICRO-80 November 1980 (Issue 12, page 10), enables me to type in lower case letters, so everything in this department seems okay.
>
> Now the problem is if I try to load the MICRO-80 Lower Case Drivers (May, 1981) I get the message followed by four and a half lines of garbage on the screen, followed by the second prompt *? I type "/" then ENTER, the screen gives a flicker, it then sits there and does nothing. The writing stays on the screen and the only way to get any more keyboard action is to power down the computer.
>
> So this brings me to a few questions. Oh! by the way, my computer is a

Level II 16K TRS-80 (cassette based) "G" board.

1. Is my tape faulty?
2. Is this mod compatible with yours?
3. If not, could you send me info on how to make it compatible?
4. If the tape isn't faulty and my mod is compatible with yours,
   H - E - L - P!!!

Keep up the good work on the best magazine out for '80 users. Long live the '80!! "

What would you say is the problem? Of course, it had to be a faulty tape; we'll send the man a new cassette but, as it was late in the afternoon, that would have to wait until tomorrow. For some reason, which now escapes me, I decided to take the tape home and check it on my System 80 - a stupid idea, upon reflection, as my internal cassette deck of early vintage is extremely fussy and will fail any tape that has the slightest flaw. But, lo and behold! Each dump loaded perfectly!

More alarming than that was the fact that when executed, the results were exactly as reported by Mr. Wilson in his letter. For the record, my System 80 was fitted with a MICRO-80 lower case modification over two years ago and which has recently been replaced by a nameless brand because I now prefer an underline cursor. After thoroughly investigating the software aspect of this dilemma, which is where I assumed the trouble was, I took a long, hard look at the installation instructions and discovered to my surprise: NOT ALL LOWER CASE MODIFICATIONS ARE EQUAL!

The original design of the TRS-80 and System 80 used a video RAM block that was only seven bits wide with bit 6 missing. Instead, hardware was used to generate bit 6 from bits 5 and 7. The typical lower case modification involves the installation of an extra RAM chip for bit 6, a new character generator ROM (if required), a little bit of wiring on the board to enable bit 6 and a software driver to enable you to make use of the new lower case capability. I expected to find that when you write a zero into bit 6 of video RAM, you can read a zero from the same location (assuming that the chip is not faulty). A close look at the modification on p.108 in "The Custom TRS-80 and Other Mysteries" shows that what goes into bit 6 is not necessarily the data on the bus. With this modification you cannot store the values 0-31 in video RAM, even though it's now 8 bits wide.

As the MICRO-80 Lower Case Driver loads into video RAM (explaining the "garbage" which is really a machine language program) and executes from there to relocate itself to the top of memory, this situation is disastrous. No wonder the system hangs and needs to be powered down. The solution is to perform the other modification given in the same book on p.107 which simply connects the new chip to bit 6 of the data bus and allows video RAM to be used just like any other RAM. Try the following program to see what happens in the video RAM of your '80:

```
10 CLS:AD%=15360
20 FOR I=0TO255
30 PRINT@896,"        ";
40 POKE AD%,I
50 IF PEEK(AD%)<> ITHENPRINT@896,"IN=";I;"OUT=";PEEK(AD%);
60 FORJ=1TO200:NEXTJ
70 NEXTI
```

In the top left-hand corner of the screen, the character corresponding to the value OUT is displayed and at the bottom you can see if your video RAM is 7 bits wide (able to store only 32 - 95 and 128 - 191), funny 8 bits (able to store only 32 - 255) or true 8 bits (able to store 0 - 255).

When it comes to hardware you must be careful not to make any false assumptions. If some problem develops then you must check very carefully the hardware changes and any software you are trying to use. It is interesting to note how, in this case, the combination of hardware and software led to such a subtle problem. To accommodate lower case ASCII characters, the video RAM must be able to store values in the range 32-127. Values in the range 0-31 (control codes) are not normally stored but are acted upon by the video driver routine. Therefore, this particular modification made no provision for storing control codes. In writing the universal Lower Case Driver routines, Eddy sought to provide maximum flexibility by allowing you to load them at any time without disturbing any other utilities or programs already in memory. The logical place to put them was into the video RAM and to relocate them from ther to free high memory. So the MICRO-80 lower case modification provided a true 8-bit video RAM which would allow you to store a m.l. program which could be executed from there (making video RAM Just like the rest of your RAM).

- 0000000000 -

***** FORM THREE *****

For our Model 3 readers, 'Desert Chase' and 'Ordering Priorities' are two Level 2 programs that should run with no problems but the machine language utility 'Formation' will definitely not run without modification. However, changes similar to those described in the October '82 Model 3 Movie Microbug (p.20) should enable you to use this utility on your Model 3.

Why don't Model 1 machine language programs always work on a Model 3? The two main reasons are that the Model 3 ROM takes up more user RAM from 42E9H to 43E8H and that the Model 3 has an extra 2K of ROM to provide extra features. The extra ROM and different hardware architecture of the Model 3 required changes to the original 12K ROM found in the Model 1. This means that some ROM calls and entry points are different on the Model 3, preventing many Model 1 m.l. programs running on the Model 3. I compared the contents of the ROM in our Model 1 to that in our Model 3 for addresses in the range ØØØØ-2FFFH and the results are set out below (Note that this may depend on the particular ROMs present):

ØØØ3 - ØØØ4	Ø1F1 - Ø1F1	124C - 124D
ØØØE - ØØØF	Ø1F3 - Ø1F4	1918 - 1918
ØØ47 - ØØ48	Ø1F8 - Ø2ØF	191C - 191C
ØØ5Ø - ØØ62	Ø212 - Ø231	1B5D - 1B5F
ØØ66 - ØØ7Ø	Ø235 - Ø245	2Ø6D - 2Ø6D
ØØ82 - ØØ82	Ø247 - Ø25E	2Ø73 - 2Ø73
ØØAA - ØØAA	Ø264 - Ø282	2Ø75 - 2Ø75
ØØAE - ØØAE	Ø284 - Ø2A7	2Ø77 - 2ØB8
ØØB2 - ØØB4	Ø2E2 - Ø2E4	2ØBC - 2ØBC
ØØC6 - ØØC6	Ø3C2 - Ø3E9	2ØF7 - 2ØF7
ØØEA - ØØEA	Ø3EB - Ø468	213B - 213B
ØØFF - Ø1Ø1	Ø46B - Ø494	2167 - 2167
Ø1Ø6 - Ø1ØA	Ø496 - Ø49E	2B85 - 2B88
Ø1ØD - Ø1ØF	Ø4AØ - Ø4B7	2B8C - 2B8E
Ø112 - Ø115	Ø4B9 - Ø5ØC	2B91 - 2B93
Ø118 - Ø11B	Ø5ØE - Ø532	2C1F - 2C42
Ø11D - Ø121	Ø534 - Ø5CF	2C7A - 2C7F
Ø125 - Ø12C	Ø5D1 - Ø5D3	2C81 - 2C82
Ø1DA - Ø1EF	Ø674 - Ø7Ø7	2C8A - 2C8C

Model 3 vs. Model 1 ROM differences

The table above shows only the addresses (in Hexadecimal) where the Model 1 and Model 3 ROMs differ. Notice that the bulk of the changes are in the first 2K of the ROM (where the I/O driver routines reside) while changes to the BASIC interpreter itself are relatively minor. Although many entry points for common routines have been maintained, the Model 1 Ø66CH entry to BASIC can no longer be used (the entry 1A19H should be used instead).

Adapting Model 1 m.l. programs is then a matter of changing ROM calls and entry points where they differ and moving those programs that reside at the bottom of BASIC's memory up to clear the extra scratch pad RAM area. With the aid of an Editor/Assembler and the source code listings in the magazine, most Model 1 m.l. programs can, with a little effort, be made to run on your Model 3. If you have any additional information or a patch that you have developed, send it in to us and we will include it in this section for the benefit of all our Model 3 readers.

- 0000000000 -

***** WHAT YOU HAVE MISSED *****

Set out below is a list of some of the programs published in early issues of MICRO-80 magazine. Back issues are available for $2.50 each or at the annual subscription rate for 12 or more copies. Cassette editions are available for all issues for $4.00 each whilst disks are available for all issues from September 1981 onwards. For 12 or more magazines with cassettes/disks ordered at the same time, the relevant annual subscription rate applies. Programs for the Hitachi Peach/ TRS-80 Colour Computer were first published in the April 1982 issue.

Issue 6 - * May 1980

* Some issues incorrectly labelled April.

SUB ATTACK (L1)
SPACE DRIVE (L1)
TRIG/BAS (L2)
SUPER SIZZLER (L2)
TIC-TAC-TOE (L2)
HOUSEHOLD ACCOUNTS (L2)
CONNECTA and CONNECTX (L2)

Issue 15 - February 1981

PINBALL (L1)
KEYNOTE (L1)
LEVEL II TBUG UPDATE (L2)
MICROHEX (L2)
SEA WOLF (L2)
ASTRONOMY (L2)
MURDER (L2)

Issue 18 - May 1981

ARITHMETIC (L1)
SORTING (L1)
NORMAL DISTRIBUTION (L2)
DISASSEMBLER (L2)
12 HOUR CLOCK (L2)
BONES (L2)
PHILATELIC ADVISER (L2)
UNIVERSAL LOWER-CASE DRIVER ROUTINES (L2)

L1 - Level 1 program	CC - Colour Computer
L2 - Level 2 program	HP - Hitachi Peach

The following back issues of MICRO-80 magazine are still available:

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
'79	-	-	-	-	-	-	-	-	-	-	-	✓
'80	✓	✓	✓	x	✓	✓	✓	x	✓	x	x	✓
'81	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
'82	✓	✓	✓	✓	✓	✓	✓	✓	✓	x	-	-
'83	-	-	-	-	-	-	✓	✓				

(- means never published, ✓ means issue available, x means issue out of print).

- 0000000000 -

PROGRAMMING

***** THEORY AND TECHNIQUES OF SORTING - PART 8 *****

by Bernie Simson

In Part 7 of this series, it was shown how records from an input file for sorting were extracted, sorted in main memory and stored away in a workfile for later merging. This article will examine in more detail the processes involved in storing sorted sublists of records in workfiles, and some merging methods.

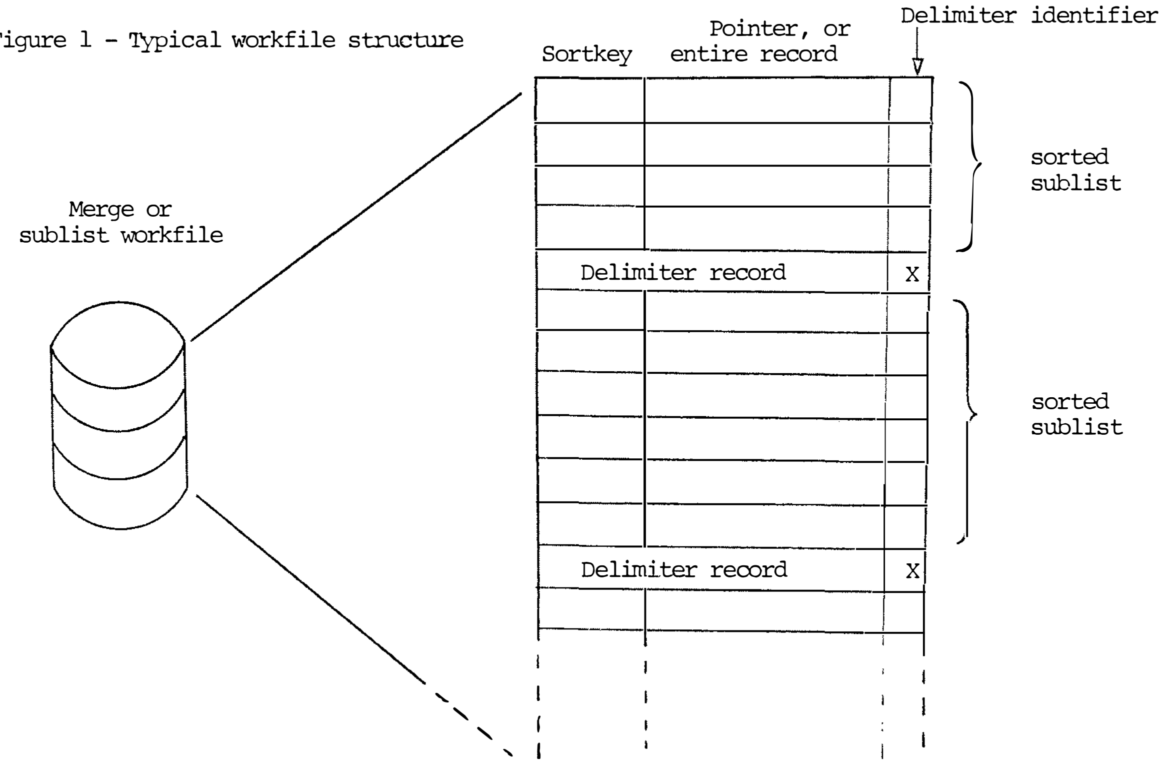
PRODUCTION OF SORTED SUBLISTS.

Since the external sort phase (merging) involves a great proportion of the total sort time due to disk operations, it follows that to optimise the sort, the total number of disk operations should be kept to a minimum, in particular, track seeks. This occurs when the read/write head is instructed to find or write a record based on a random address, rather than a read or write of the next sequential record. This is not always an easy criterion to allow for in a restricted disk-based system, which is usually the case with most microcomputer configurations.

A restricted disk system is defined as a system having less than four head actuators. A head actuator refers to the mechanism in a disk drive that controls the movement of the read/write head, notthe read/write head itself.

The reason for this is explained later; a system having four logical drives in two double sided disk drives has only two head actuators, and is therefore considered a restricted disk system

Figure 1 - Typical workfile structure



for optimum merge time purposes.

When a sublist has been sorted in memory and is ready for storage in a workfile, the records (or ADDROUT pointers) are written out with the same sortkey used to determine ordering in the internal sort phase. This same sortkey is used to determine ordering during merging. Now, depending on which variation of merge technique is used, which is governed by the number of head actuators available, the sorted sublist is written out to one workfile, or several different workfiles. Also, a sublist delimiter is written at the end of each sublist to separate them in the same workfile. The delimiter would obviously have to contain some unique identifier to avoid it becoming confused with a sublist record. Figure 1 shows the format of a typical workfile.

If more than one workfile is used to store the sublists, then they are used in an alternating fashion, so that the sublists are distributed fairly evenly over the workfiles.

When all the records from the input file have been extracted and sorted in memory, the result should be one or more workfiles containing sorted sublists separated by delimiters. The workfile(s) are then used as input to the merge process.

MERGING TECHNIQUES.

This is where most of the disk operations occur, so it is wise to devote some extra thought to optimising this aspect of the sort if you are designing a sort package.

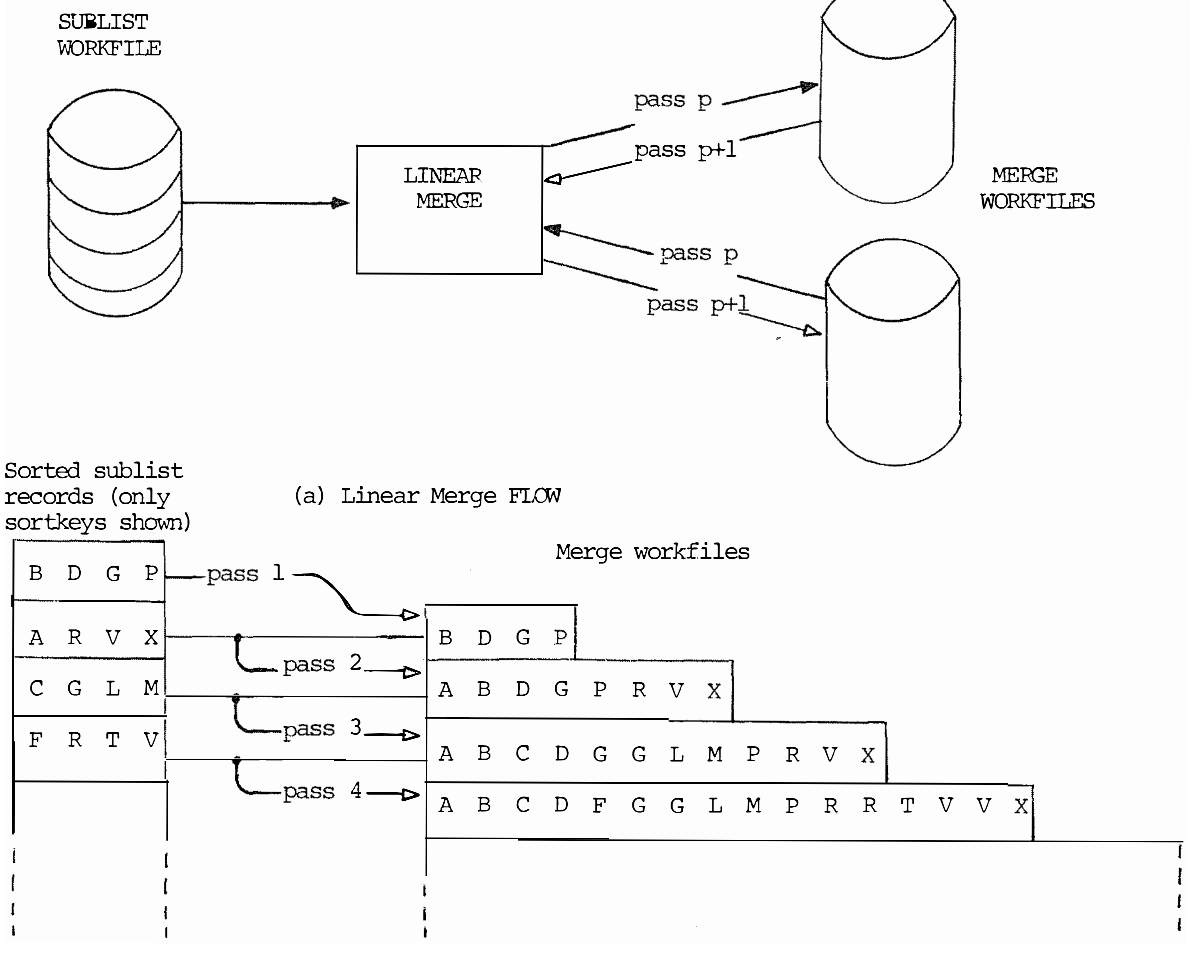
Two merge techniques will be shown, with some variations applicable to restricted disk systems. The two merge types are:

Linear Merge, and
Tree Merge.

LINEAR MERGE.

Linear merging is shown in Figure 2. Only one sublist workfile is required as input. It involves reading the sublist records and merging them with records in the merge workfile that was created

Figure 2 - Linear Merge



(a) Linear Merge FLOW

(b) Linear Merge TRACE

during the previous merge pass, to produce a new merge workfile. This is then used as input to the next merge pass, together with the next sublist in the sublist workfile. The merge workfile output from each merge pass grows in a linear fashion.

Do not confuse sublist workfiles with merge workfiles. Sublist workfiles are created when sorted sublists are stored on completion of each sublist memory sort, whereas merge workfiles are temporary files created by the merge process. The internal operation itself is quite simple, and is common to both merge types. At any one point in time, two records from each file to be merged are compared, and the one with the smallest sortkey value is written out to another merge workfile, and the next record from the workfile from which the written record came, is read, and the comparison repeated. A three-way merge could be implemented if desired, involving reading 3 files at one time, and writing out a record depending on the result of a three-way comparison, but this discussion is restricted to a two-way merge to avoid confusion (if you aren't confused already).

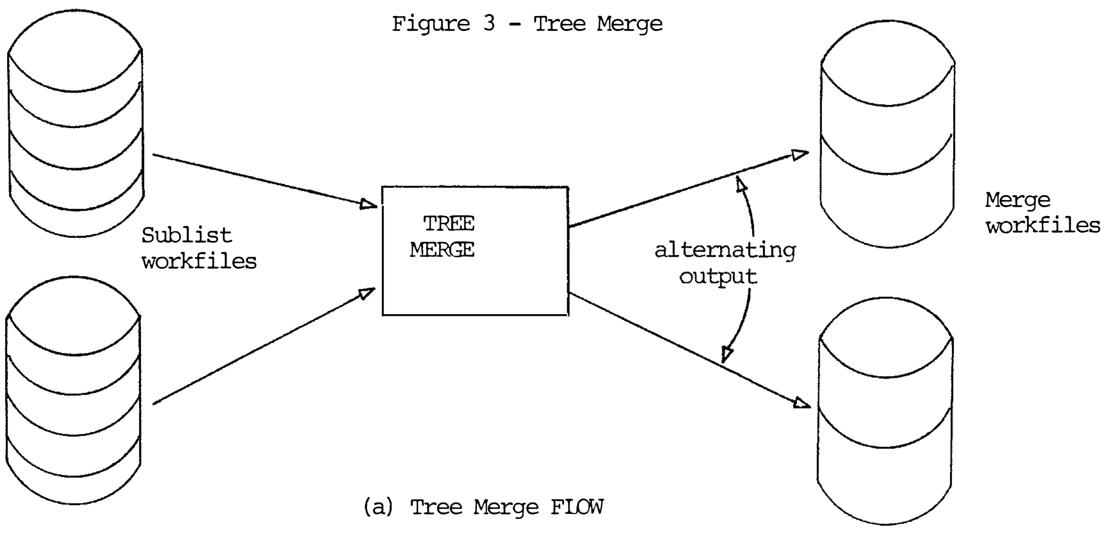
A variation to this merge technique involves reading all the records in the sublist workfile into memory, and using this array as one of the inputs to the merge (with the merge workfile). This assumes that the entire sublist will fit into memory. It should do so, as it was originally written out from memory in the internal sort phase, unless the sublist was produced according to the process described in Figure 2(A) of the previous article (Part 7). You will notice that in this case, the original input file is referenced to retrieve the entire record using the pointer in the array as index, thereby creating a sublist in the workfile that is larger than the sorted array.

This variation is useful where only two head actuators are available - one to read the merge workfile, the other to write the new merge workfile, while actual merge is taking place. Track seeks will not occur, even though 3 files are involved, because the reading of the sublist workfile is not simultaneous with the reading and writing of the merge workfiles.

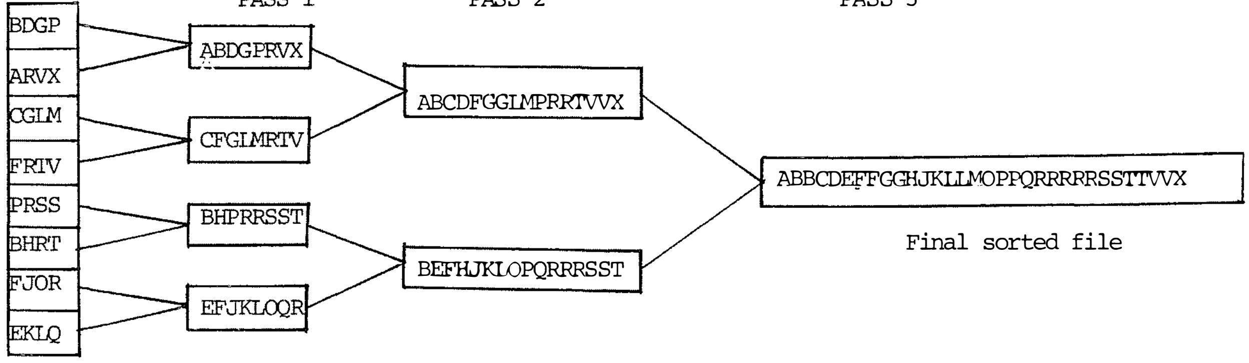
TREE MERGE.

Tree merging is shown in Figure 3. Although the sorted sublists are distributed over 2 sublist workfiles, the structure of each is still consistent with Figure 1. However, you will notice the different algorithms as evident in the merge pass traces in Figure 2(B) and 3(B). The tree merge creates multiple merge workfiles, again distributing them over another 2 new merge workfiles

Figure 3 - Tree Merge



(a) Tree Merge FLOW



(b) Tree Merge TRACE

in an alternating fashion in the same way that the sublist workfiles were produced. These 2 new merge workfiles are then used as input in the next pass to produce 2 more merge workfiles, this time replacing the 2 merge workfiles used as input in the previous pass. This technique produces sublists that are continually doubling in size, until only 2 sublists exist in separate workfiles, when they are finally merged to produce the final sorted file.

Now in order to eliminate track seek, the four workfiles active during any one pass must exist on separate disks under the control of separate head actuators. A double sided drive has only one head actuator, so unless 2 workfiles are placed on the same tracks on each side of the disk, track seek will occur. I don't know of any DOS that will allow, without complexity, control over where on the disk a file is placed.

If less than four head actuators are available, then it may be better to have only one sublist workfile, as in the Linear merge, rather than incur an overhead of two extra files on the disk. Track seek will occur anyway.

If this variation is adopted, two sublists are read simultaneously for merging, so it will be necessary to maintain some information as to the starting address of each sublist, maybe by using a linked list that is updated when the sublist workfile is created. The output from this variation is also just one workfile, which is used as input for the next merge pass. The algorithm is the same as for the four-file Tree merge - doubling the sublist sizes on each pass.

COMPARISON OF THE TWO MERGE ALGORITHMS.

The Linear merge is the simpler to implement, but the Tree merge is the more efficient. This is usually the case in life....decisions, decisions.

The sort package designer must decide on a trade-off between simplicity or efficiency, also taking into account the environment in which the sort package will be implemented, in particular, the likely number of head actuators available.

To assist the budding designer, I have prepared a table giving the maximum number of sublists serviced by one head actuator during a merge pass, for the two algorithms. The smaller the number, the less track seek activity.

MERGE TYPE:		LINEAR	LINEAR VARIATION	TREE	TREE VARIATION
NO. HEAD ACTUATORS	4	1	1	1	2
	2	2	1	2	2
	1	3	2	3	3

This gives an indication of the likely track seek activity for a given environment, which will have a bearing on the overall efficiency of a particular algorithm - for instance, the Tree merge algorithm, although being more efficient than the Linear variety as demonstrated below, may turn out to be less efficient if bogged down in a multitude of track seeks in each pass.

EFFICIENCY ANALYSES.

For the purposes of these analyses, it is assumed that track seek activity is the same for both algorithms, and therefore does not enter into the comparisons.

An accurate measure of the efficiency of a merge algorithm can be made by examining the average number of times any particular record in a workfile is moved from the first pass to its placement in the final sorted file. That's what merging is all about - moving records from one file to another.

This can be done by determining the number of passes that will be necessary to merge all the sublist records, governed by:

N...	The number of records in the original input file (and therefore, the number of records in all the sublists),
S...	The size of each sublist, which is governed by the internal sort array size (i.e. memory size).

Then the total number of records output in all the passes is calculated, and finally, this is divided by N to give the average times each record is moved.

LINEAR MERGE.

From the Trace in Figure 2(B), it is evident that each pass involves writing 4 more output records than in the previous pass, i.e. the output merge workfile grows by the size of the input sublist.

If you were to list the number of output records for each pass, you would notice that the numbers represent an arithmetic progression, where the sublist size is the first term, and also the common difference.

So, substituting the variables in the arithmetic progression formula:

    Sum = (N/2)*(2*A + D*(N-1))

Where N = number of terms, A = First term, D = common difference, we get:

    Sum = (N/(2*S))*(2*S + S*((N/S)-1))

Because the number of passes = N/S (Also the number of terms in the progression), the first term is S, and the common difference is also S.

This formula can be simplified to:

    Sum = N*(S+N)/(2*S)

This represents the total number of records output in all the passes, so the average number of records moved is:

    Ave = (S+N)/(2*S)

Now for the Tree merge analysis.

TREE MERGE.

From the trace in figure 3(B), a relationship can be seen between the number of passes required to produce the final sorted file, and the number of sublists in the workfile(s) at the start of the first pass. This relationship is exponential, and is defined by:

$$2^P = L$$

Where P = number of passes, and L = number of original sublists. Since L is defined by N/S (total number of records in the original unsorted file divided by size of each sublist), the number of passes is derived as:

    P = LOG(N/S)/LOG(2)

or LOG(N/S) to Base 2.

For the purposes of this analysis, it is assumed that the number of sublists before the first pass is a power of 2, i.e. 2^3. This is where the Tree merge algorithm becomes more complex to implement, when the number of sublists is not a power of 2, so that during merging, a sublist is left unpaired and must be kept track of for merging in the next pass.

The relationship in Figure 3(B) is very similar to the number of accesses required to traverse a binary tree or to do a binary search, to locate an element. Since in each pass, each sublist doubles in size, and the number of sublists are halved, the total number of output records written in each pass remains the same as N (total number of original records). Therefore, the total number of records output is:

    N*(LOG(N/S) to Base 2)

and the average number of times each record is moved is:

    Ave = LOG(N/S) to Base 2,

which also happens to be the number of passes, a fact that is evident from the trace in figure 3(B).

Now that the formulae have been derived for the 2 algorithms, their efficiencies can be compared, given different sort file sizes, and sublist sizes (i.e. memory sizes). Using the formulae, the average number of times each record is moved in the entire merge is:

N (FILE SIZE)	S (SUBLIST SIZE)	LINEAR MERGE	TREE MERGE
256 recs	64 recs	2.5	2
1024	64	8.5	4
1024	512	1.5	1
16384	512	16.5	5
50000	2000	13	4.6
100	100	1	0

The figures obtained in the last line are correct according to the formulae. However, in practice, where the file size is the same as the sublist size (entire file fits in memory), the Linear merge will not transfer the sublist to a workfile, as shown in the trace in Figure 2(B).

Obviously, if all the file fits in memory, no merge is necessary, so a figure of zero means no merge, or record movement.

This comparison demonstrates the efficiency of Tree merge over Linear merge but, as mentioned above, it may be affected by other factors such as track seek, so trial and error may be the only way to determine which is most suited to a particular environment.

TO SUMMARISE...

Merging is the final process involved in sorting a file of records. Its purpose is to combine sorted sublists produced by the internal sort process because the input file is too big to fit in main memory at the one time.

The placement of sorted sublist workfiles in a restricted disk-based system is important in order to optimise sort time due to the effect of track seek activity. Therefore, some trade-offs may be necessary to minimise this.

Two merge algorithms are Linear Merge and Tree merge. Tree merge is the more efficient in a system where track seek activity is not a limiting factor. Where it is, the choice as to which will be more efficient is governed by the particular environment where the sort package is installed.

- 00000000000 -

***** MICROBUGS *****

Although we make every effort to ensure accuracy in the material we publish, inevitably errors and omissions will occur. In this section, we print corrections to those bugs that have been reported.

GOLF - July, 1983 Vol.3, No.12 - page 27.

The listing published in the magazine contains a small error in line 2170. It should read:

```
    217Ø IFS1<TPARTHENS1$="UNDER":SX=TPAR-S1ELSES1$=" =PAR":SX=S1
```

The cassette version of the program also contains this error.

- 0000000000 -

SOFTWARE

***** THE TOWERS OF HANOI - (Colour) by M. Byrne *****

The 'Towers of Hanoi' is played with a number of disks of differing sizes and three pegs on which the disks may be stacked. Initially the disks are arranged on one peg in order of decreasing size. The object is to transfer them (in the least number of moves) to another of the pegs so they are once again arranged in order of decreasing size. However, you may only move one disk at a time and it must not be placed on a smaller disk.

This program allows you to select the number of disks (up to a maximum of 10 for the Hitachi Peach and 8 for the Colour Computer) and whether you or the computer will solve the problem.

The main variables used are:

N(3)	holds the number of disks on each peg.
P1(), P2(), P3()	are peg arrays. Each element holds the size of the disk at that position.
SK()	the parameter stack.
SC	the source peg.
DN	the destination peg.
AL	the alternate peg.
NM	the move counter.
SP	the stack pointer.
B	the base pointer for the stack elements of the previously invoked procedure.
CT	the number of disks.
S	the amount of delay (for automatic operation).
F	the peg the disk is moved from.
T	the peg the disk is moved to.
T1, T2	used to test whether the move is legal.
M	is the minimum number of moves.
TD	the top disk of the peg we are moving from.

Probably the feature which will cause most confusion is the use of recursion. It appears to be a not-too-well-known fact that BASIC will allow recursion (on most machines anyway). Recursion is a means whereby a procedure or subroutine may call itself. This involves the use of a stack (which BASIC provides) for storing the return address and another stack (which must be set up explicitly) if parameters are to be passed to the called procedure.

The parameter stack is only really used when automatic operation is required as manual operation merely involves asking for moves and checking their validity.

The current top-of-stack is indicated by SP. When used by the main procedure, the top stack element holds the number of the alternate peg for this move, top of stack - 1 holds the number of destination peg for this move, top of stack - 2 holds the number of the source peg for this move while top of stack - 3 indicates the numbers of disks still to be moved.

When used by the shift procedures the top of stack contains the number of the destination peg and top of stack - 1 contains the number of the source peg.

Because the parameter stack must be explicitly set up and manipulated in BASIC, the algorithm tends to become somewhat obscured. Also if it is the first time you have encountered recursion, BASIC is not the ideal language to learn about it. For those of you who are interested in finding out more, the tree-diagram for the algorithm used in this program is given in the book 'An Introduction to Problem Solving Using Pascal' by Kenneth Bowles.

- 0000000000 -

***** OTHELLO - Colour Computer by S. Gibbons *****

Othello is written for the 16K Tandy colour computer. Othello is a game of strategy for two players and is based on, and very similar to, the board game by the same name.

The rules are the same as the original and are explained in the program.

The playing board is made up of 64 squares (8 x 8). The vertical and horizontal positions are represented by numbers 1-8 shown at the top, bottom and both sides of the board.

The disks are represented by squares which are black on one side and white on the other. (Only one side of the disk is shown at any time).

The disks are placed alternately by each player starting with black. To place a disk on the board you give the horizontal position then a comma followed by the vertical position. The computer will tell you if you give the coordinates of an illegal position.

The progressive score is constantly shown at the top right hand corner throughout the game. The score is based on the amount of disks each player has on the board in his possession.

The game will end when all the board has been filled with disks. The winner is determined by the colour which has the most disks.

- 0000000000 -

***** REGISTER DISPLAY PROGRAM (Peach) - by D.J. Omond *****

This provides an easy means of examining register contents at any time during the operation of a program. It is similar to the facility provided by the machine-language monitor but whereas that can only be initiated from the keyboard (MON + R), the Display Program can provide register content information at any point in a program then can revert either to the program under test, to BASIC command level or to the machine-language monitor.

LOADING.

A short program written in BASIC loads the data listing, sets the routine into operation to first initialize the Software Interrupt vector then provide a sample display. At this stage, the BASIC loading program can be discarded.

The location at which the operational program is stored in memory can be specified by the operator if desired during the BASIC loading phase. Alternatively, the program can be stored in a "default" location at the top end of User RAM.

OPERATION.

The display program is initiated by a Software Interrupt i.e. by a 3F instruction inserted in a program under test. Alternatively, an EXEC command inserted in a BASIC Program and directing control to the display routine, provides return options as follows:-

(a) EXEC &H6F95 returns to the program under test

(b) EXEC &H6F97 returns to BASIC command mode

(c) EXEC &H6F9B returns to the machine-language monitor

On the occurrence of a SWI, all registers are saved on the stack. The stack pointer value is derived (6F9F) and placed on the stack. Use is made of various subroutines in the ROM to designate and display the register contents e.g. 6FA7 prints 'S' then 6FAC-6FB3 the SP value. 6FB9 refers to the register headings table (used by MON) then up to 6FCA, displays registers CC, A, B, DP in successive loops.

6FD0 to 6FE1 prints the 2-byte registers X, Y, U and PC and 6FCC-E tests for completion. 6FE3-F is a stack of "print spaces" which is entered at various levels to provide the required number of blanks for the screen display.

The last two lines 6FF3-5 restore all register values (equivalent to RTI) following which control moves to the PC value stored at the time of Software Interrupt. This is the means by which the return options are tied to the point of entry into the display routine.

- 0000000000 -

***** DESERT - CHASE - L2/16K by A.N. Downey *****

In this game of high adventure, the object is to travel a distance, which varies between 200 and 300 miles, across the vast Simpson Desert.

Your journey is made perilous due to hazards such as: wild tribesmen, lack of water, sandstorms, etc. As all good travellers of the desert know, you must take care of yourself and your equipment. However, if you find yourself in trouble you can always hope for help - although this is not always forthcoming.

The game is extremely easy to play and is very addictive. Many hours have been spent at my keyboard trying to beat the odds. There is some strategy involved in this game, however, with the number of random variables included it is extremely difficult to rely upon this fact.

As for the program itself: all line numbers are in intervals of 10, beginning at 10 so "AUTO" may be used through the program. I apologise for the lengthy graphic lines (Line No's 990 - 1430). However, I am sure some of you can design your own routine for these. I assure you that, although lengthy, they are well worth the work involved. The distance required to travel is set up in variable R - contained in Line No. 20. This may be changed to suit your requirements (make it less for a shorter distance).

I won't explain any more as it will take some of the fun out of it, and the program is self-prompting. So, good luck with your Desert Chase.

- 0000000000 -

***** FORMATION - L2/16K (C) by David Grigg *****

An Electronic Form Creation and Data Entry System.

INTRODUCTION.

Entering large amounts of data can be the most tedious and frustrating part of using a personal computer. Storing that data on cassette tape, if you don't have disk drives, can be even more time consuming and annoying. If data is to be entered during program execution, then appropriate prompts can be written into the program in the form of PRINT or INPUT statements, but this can be fiddly and difficult. Data entered in this way can only be stored on tape using the snail's-pace PRINT # statements, and what's more, there is always the chance that a program error will cause execution to stop, which usually means losing data entered during execution. Rather safer is entering data in the form of DATA statements in the program itself. But having to type the word "DATA" at the beginning of every line can be tedious. Commas have to be inserted between data items, and it is all too easy, with a long list of numerical data items, to lose track of how many you have entered, or which order they are in, since there are no prompts. So this method of data entry can be equally tedious.

In a non-computer situation, we are all used to filling in forms when information is wanted. Although this can be wearying, at least we always know what data is wanted, how much room we have to fill it in, and we can always go back to correct a previous entered item, if we have made a mistake.

Business computer systems have offered electronic form systems for data entry for some time, whereby an electronic form is presented on screen so that a relatively unsophisticated operator can enter data in the same sort of way as filling in a form by hand, with all the advantages of that method.

FORMATION is a computer program for the Model 1 TRS-80 and the System 80 which gives you the same capability, with the very valuable feature of being able to quickly and simply design your own electronic forms, then save the design for future use. Data is stored in the form of DATA statements automatically appended to any existing BASIC program in memory. The data can then be stored on tape simply by using the CSAVE command in the normal manner.

HOW TO USE "FORMATION".

1. Loading the program.

   Turn on the computer then type SYSTEM (enter). To the SYSTEM prompt answer FORMAT (enter), and press the play button on the recorder. When the prompt returns, type / (enter), and the READY prompt will appear. Now you can load any BASIC program you wish to append data to. When you are ready to use FORMATION, type OPEN in command mode. The list of FORMATION options should appear.

2. A Sample Form

   To see how FORMATION works, choose option 4, FILLING EXISTING FORMS. The screen will fill up with a sample electronic form, for a simple mailing list program. Look at the format of the screen. The areas marked out by underline symbols ( ) are "entry fields". A blinking block cursor will be present at the beginning of the first field.

   Type in some appropriate data, perhaps your own surname first. Then hit ENTER (NEW LINE for the System 80). The block cursor will skip to the beginning of the next entry field. The down arrow key would have the same effect (CTRL key for System 80). Before you enter your initials hit the up arrow key (or ESC Key). The cursor will skip back to the beginning of the previous field. Hit ENTER again to go forward to the next field. Continue to fill in appropriate data in all the fields. If you make a mistake, backspace in the usual way, or use the up arrow key (or ESC Key) to return to a previous field. Once all the data is entered to your satisfaction, hit CLEAR (or for the System 80, hit shift @). All the data will vanish, and the cursor will be back at the beginning of the first field.

   Note that if you type right to the end of a field, the cursor automatically skips to the next field. Similarly, backspacing past the beginning of a field skips you back to the start of the previous field.

   You can either continue to enter new data, or you may like to escape from this mode to see what has happened to your data. To do this, hit shift up arrow key (or shift ESC). The list of options will return. This time choose option 5, RETURN TO BASIC. The Ready prompt will appear.

   Now LIST. The data you have entered will be listed as one or more DATA statements beginning at line 32005, with commas inserted between items which were entered in different entry fields. If an existing BASIC program had been in memory, these statements would have been appended at the end of this program.

   To return to FORMATION type OPEN (enter).

3. Creating a New Form

   To create your own electronic form, choose option 2, CREATE A NEW FORM. Or the existing form in memory can be modified or added to by choosing option 1. If option 2 is chosen, the screen will clear, and the cursor will appear at the top left of the screen.

   You may now type in any text you choose to act as prompts. The down arrow key (or CONTROL) key will drop the cursor down one line. The up arrow key (or ESC) key will jump it up one line. The backspace key acts normally, though you will find that the cursor is non-destructive. If you want to wipe out a character, you should use the space bar. On the TRS-80, the right arrow key acts as a non-destructive forward space. The ENTER (or NEW LINE) key will move the cursor to the start of the next line. Using these keys, the cursor can be quickly moved around the screen to enter text wherever you like.

   However, to create the vital entry fields, the underline symbol must be used. This symbol is accessible by hitting the @ key. An entry field begins with an underline symbol and continues to the end of a string of such symbols. Any break in this string creates a new entry field. So if you were to type "   " this would establish one entry field four characters long, but typing "   " creates four entry fields, each one character long. Entry fields can be created anywhere on the screen, and you may create up to 80 separate entry fields.

   It will be important for you to make a written record of which entry fields you have established, what data they will contain, and their order (from left to right, top to bottom). The reason you need to note this is so that your BASIC program will be able to access the data in the right order, and with the right type (numerical or string as appropriate).

   Once the form is complete to your satisfaction, type shift up arrow key (or shift ESC) to return to the option list. If your form is only for temporary usage, you can proceed to fill this form as you did with the sample form. However, you will probably wish to save this form for future use. To do this, put a blank cassette in your recorder, and choose option 3. After you have pressed the record and play buttons, you can save the form onto tape.

Note that the entire FORMATION program, complete with new form, is saved in this way. This means that any time you wish to use your own form, you need only load the one SYSTEM Program. FORMATION was designed in this way for your convenience.

NOTE: Some characters, if entered when filling in forms, will be changed before they are inserted into the DATA statements. This is because otherwise they would act as delimiters and mess up the DATA statements. These characters are double quotes ("), comma (,), and the colon (:). They will be changed to the characters ↑, ↓ , and → respectively. Your BASIC program may need to take this into account.

FORMATION COMMAND SUMMARY.

CREATE MODE:	TRS-80	SYSTEM 80
* Move cursor up one line	↑	ESC
* Move cursor down one line	↓	CNTRL
* Move cursor forward one space	→	not available
* Move cursor back one space	←	←
* Insert underline symbol	@	@
* Move cursor to start of new line	ENTER	NEW LINE
* Escape from this mode	shift ↑	shift ESC
FILL MODE:		
* Cursor to start of last field	↑	ESC
* Cursor to start of next field	↓	CNTRL
* Move cursor forward one space	→	not available
* Move cursor back one space	←	←
* Move cursor to start of next field	ENTER	NEW LINE
* Record data, start clean form	CLEAR	Shift @
* Escape from this mode	shift ↑	shift ESC

FROM BASIC:

Type OPEN (enter) to return to FORMATION

NOTE: FORMATION makes extensive use of calls to the Microsoft Basic ROM in order to reduce the amount of memory it takes up. It is possible that the program will not work with some late model TRS-80's with the revised ROM.

This program is for L2/16K machines only. Users of Level 2 machines with more than 16K MUST respond to MEMORY SIZE with 32767. This program WILL NOT work on a Disk Basic machine.

SYSTEM 80 owners who do not have a right arrow key may get around the problem of right direction cursor movement by pressing the backspace key until the cursor wraps around the screen. The desired position may then be reached as it is approached from the right side of the screen.

(The distribution disk contains an offset version taking the user to Level 2).

- 000000000 -

***** PRIORITIES - L2/16K by A.F.J. Bell *****

This program was written for two reasons. Firstly, because of the activities of the Razor Gang there was a need to be able to review the activities of my hospital and put them in order of priority. Secondly, I wanted to get some experience in using string matrices.

The program enables one to specify up to 12 activities, and to assign a rank to the importance and to the feasibility of each activity. Then the computer finds the product of the importance and the feasibility, which can be taken as the relative priority which each activity should receive. One can review, add or change data until one is satisfied. One can then order the activities according to their importance, feasibility, or priority, and printout the results.

Lines 10 to 120 are housekeeping. Line 60 sets my Microline 80 Lineprinter to 10 chrs/in and 64 chrs/line, sets the page length to A4 size, and sets the linenumber counter to 0. You will need to change this to suit your type of printer.

Lines 190 to 240 display the options. The rather complicated system in lines 220 to 240 enable one to use a single stroke INKEY$ system.

CHR$(13) in lines 250 to 300 (the instructions) is the control character for linefeed/carriage return.

Lines 310 to 440 accept data from the keyboard, and puts the data as string values in a string matrix (two dimensional array). Line 430 converts the string values into numerical values,

MICRO-80 PRODUCTS CATALOGUE

This catalogue contains a selection from the wide range of peripherals, interfaces, computers and software carried by MICRO-80 for your computer. If you don't see the item you want, contact us, we probably have it anyway!

MICRO-80 has been supplying customers throughout Australia and the Pacific region by mail-order for 2½ years. Our customers find this a simple and efficient way to do business. You may place your order by telephone or by mailing the order form from any issue of MICRO-80 magazine. Generally, it takes about one week from receipt of order until despatch. You should allow 2-3 days for your letter to reach us and 7-10 days for the parcel to reach you, making a total turnaround time of 2½-3 weeks.

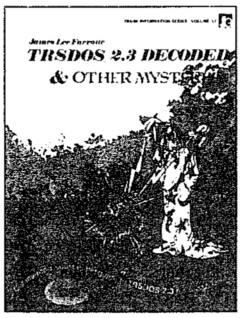
WARRANTY AND SERVICE

All hardware products carry a 90 day parts and labour warranty either from the manufacturer/distributor or from MICRO-80 Pty Ltd. In many cases, warranty servicing can be arranged in your own city, otherwise goods will be repaired by our own team of technicians in our Adelaide workshops.

TRADE-INS AND TERMS

MICRO-80 can accept your existing equipment as a trade-in on new equipment. We can also arrange consumer mortgage financing or leasing on larger hardware purchases. Contact us for details.

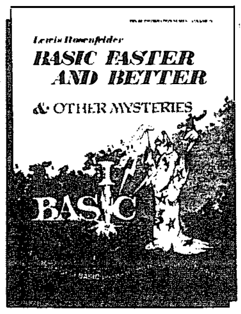
BOOKS



TRSDOS 2.3 DECODED AND OTHER MYSTERIES

by James Farnour

The TRSDOS operating system explained with a fully commented source code listing, a description of the command line interpreter and the subroutines used by the DOS commands, the file management system, the error message processor and much, much more. Excellent value at $39.95 (plus $1.20 p. & p.)



BASIC FASTER AND BETTER AND OTHER MYSTERIES

by Lewis Rosenfelder

Basic is not nearly as slow as most programmers think. *Basic Faster and Better* shows you how to super charge your BASIC with almost 300 pages of fast, functions and subroutines.
You won't find any trivial poorly designed "check-book balancing" programs in this book — it's packed with *useful* programs.
Tutorial for the beginner, instructive for the advanced, and invaluable for the professional, this book doesn't just talk . . . it shows how! Basic Faster and Better is $39.95 (plus $1.20 p. & p.)

BOOKS

THE CUSTOM TRS-80 AND OTHER MYSTERIES

by Dennis Bathory Kitsz

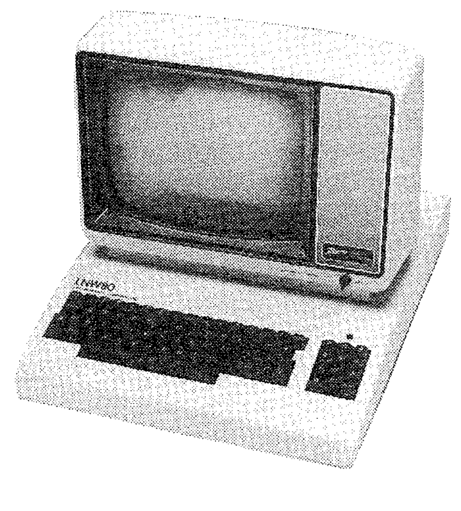
Ever wanted to do things to your TRS-80 that Radio Shack said couldn't be done? How about reverse video, high resolution graphics, and audible keystrokes?
Now enough? How about turning an 8-track into a mass storage device, making music, controlling a synthesiser, individual reverse characters, and a real-time clock just to name a few?
The *Custom TRS-80 and Other Mysteries* is packed with more than 290 pages of practical information and can be yours for only $39.95 (plus $1.20 p. & p.)

MACHINE LANGUAGE DISK I/O and OTHER MYSTERIES

by Michael Wagner

The guide to machine language disk software for TRS-80 Models 1 and 3 that explains what the floppy drive system is all about, disk controller chips, read/write access, head step and seek commands, forced-interrupt command, disk files, how records are stored on disk, creating a file, etc. You also get a disk formatter program, a program to calculate passwords and much more for only $39.95 (plus $1.20 p. & p.)

Introducing the NEW LNW80 MODEL 2



The new LNW Model 2 is not just a microcomputer but a complete computer package that includes excellent hardware, extensive systems software and a range of application software. Manufactured by LNW Research Corporation, the LNW Model 2 features:

HARDWARE:

- 4MHz Z80A microprocessor with 96K user RAM.
- 16K x 6 bits Graphics RAM, expandable to 64K (four pages).
- Printer, RS-232-C and cassette interfaces.
- 12K BASIC in ROM (Level 2 compatible).
- Support for single/double sided, single/double density, 5¼" or 8" disk drives.
- Full TRS-80 Model 1 compatibility.
- Hi-res Colour (RGB) and B & W video outputs.
- Four Hi-Res Graphics Modes:
  — B & W 128 x 48
  — B & W 480 x 192
  — 8 colour 128 x 192
  — 8 colour 480 x 192
- Three text displays Modes:
  — 64 char x 16 lines
  — 80 char x 16 lines
  — 80 char x 24 lines

SYSTEMS SOFTWARE:

LNWBASIC 4.0 — an extension to disk BASIC that allows full use of the LNW80's graphics capabilities through the addition of powerful graphics commands such as CIRCLE, COLOR, DRAW, etc.

DOSPLUS 3.4 — the fast, reliable and easy-to-use operating system that provides all the file control and disk management you need for maximum benefit from your disk drives as well as an enhanced disk BASIC.

LNW-CP/M — the CP/M operating system opens the door to a whole new world of software. LNW-CP/M was designed to be compatible with application programs written for CP/M 2.2 and provides the user with a 61K system

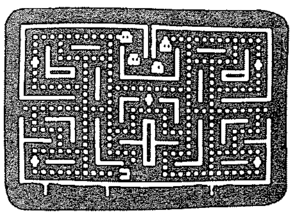
APPLICATION SOFTWARE:

1. LNW Small Business and Professional Accounting Series — *including General Ledger, Accounts Receivable, Accounts Payable and Payroll (U.S.A. conventions).*
2. Electric Spreadsheet — *for financial planning and forecasting.*
3. Electric Pencil — *an easy-to-use word processor.*
4. Microterm — *an intelligent terminal program for communications.*
5. Chart-Ex — *allows you to transform your data into pie, line or bar charts on hi-res display or printer with bit graphics.*

The LNW80 Model 2 is perfect for the serious hobbyist or businessman seeking a higher performance, more reliable computer to replace his TRS-80 Model 1 without sacrificing a huge investment in software and programming experience.

LNW80 Model 2 computer *(complete except for disk drives and monitor)*	$2,850
HI-RES Green Phosphor Monitor	$265
Super HI-RES Hitachi RGB Monitor	$1,250
Two Single-sided 40 Track Double Density Disk Drives *(in cabinet with power supply and cable)*	$825

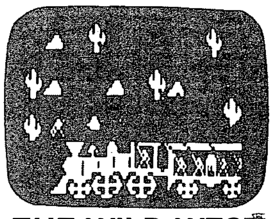
All prices include Sales Tax and are subject to change without notice. Prices are FOB Adelaide. Add $20 road freight anywhere in Australia. All equipment carries MICRO-80's Australia-wide 90 day warranty covering parts and labour.



SCARFMAN

This incredibly popular game craze now runs on your TRS-80! It's eat or be eaten. You run Scrarfman around the maze, gobbling up everything in your path. Try to eat it all before nasty monsters devour you. Excellent high speed machine language action game from the Cornsoft Group. With sound.

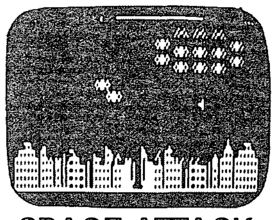
Price: $17.95



THE WILD WEST

It's up to you to keep the West beautiful with Outlaws and renegade Indians on all sides. Even the train has been captured by Outlaws with all the payroll on board. Can you clean up the Wild West?

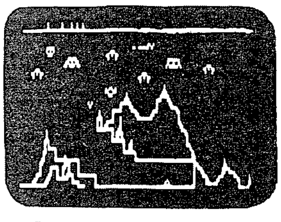
Price: $26.50



SPACE ATTACK

Steady your nerves, keep a sharp lookout, and prepare for battle to save your city. Fiendish aliens are all around, and if they destroy the city you lose.

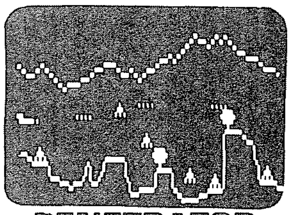
Price: $26.50



STRIKE FORCE

As the primary defender of a world of cities under deadly alien attack, your weaponry is the latest rapid fire missiles, long range radar, and incendiary "star shells." Your force field can absorb only a limited number of impacts. A complex game of strategy, skill and reflexes from Melbourne House.

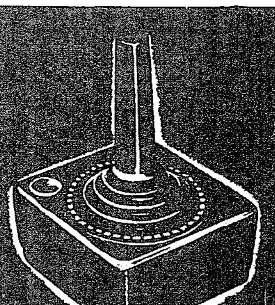
Price: $26.50



PENETRATOR

Soar swiftly over jagged landscape, swooping high and low to avoid obstacles and enemy missiles attacks. With miles of wild terrain and tunnels to penetrate, you're well armed with bombs and multiple forward missile capability. From Melbourne House. Features sound, trainer mode and customizing program.

Price: $36.50



STICKEROO JOYSTICK INTERFACE

for the TRS-80 MODEL I and SYSTEM 80

FROM **$32.00**

ADD $2.00 p. & p.

CONVERT YOUR COMPUTER INTO AN ARCADE GAMES MACHINE

Micro-80's Stickeroo Interface Features:

•Compatible with Joysticks for Atari, Vic-20 and most video games •Saves your keyboard from abuse •Compatible with programs from leading US software houses: Big Five, Cornsoft, Melbourne House, Adventure International •Adds a whole new dimension of pleasure and fun to your favourite games •Will be supported in MICRO-80 •Can be used with your own basic or ML Programs •Comes complete, ready to plug in and use •Absolutely no modifications required to your computer.

Due to popular demand, Stickeroo Interface is now available separately so you can use the Joystick of your choice.

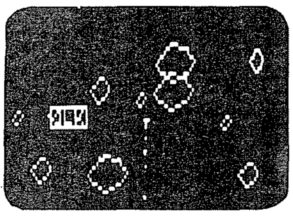
PRICE INCLUDES ... STICKEROO + INSTRUCTIONS + DEMO PROGRAM LISTING
PLEASE SPECIFY TRS-80 MODEL I OR SYSTEM 80 WHEN ORDERING

The Stickeroo Interface plugs in to the expansion edge connector and may not be suitable for expanded systems.

PISTOL GRIP JOYSTICK WITH FIRE BUTTON

S25 + S2 p & p (No p & p required if ordered with Stickeroo Interface)

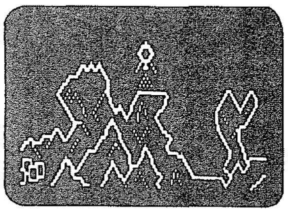
ALL GAMES ADVERTISED ON THIS PAGE ARE STICKEROO COMPATIBLE



SUPER NOVA

Asteroids float ominously around the screen. You must destroy the asteroids before they destroy you! (Big asteroids break into little ones). Your ship will respond to thrust, rotate, hyperspace and fire. Watch out for that saucer with the laser! As reviewed in May 1981 Byte Magazine.

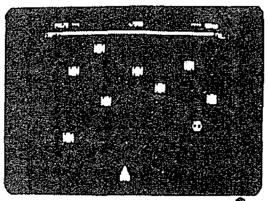
Price: $26.50



LUNAR LANDER

As a vast panoramic moonscape scrolls by, select one of many landing sights. The more perilous the spot, the more points scored -- if you land safely. You control LEM main engines and side thrusters. One of the best uses of TRS-80 graphics we have ever seen. From Adventure International. With sound.

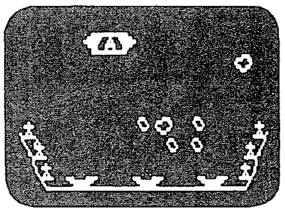
Price: $26.50



COSMIC FIGHTER®

Your ship comes out of hyperspace under a convoy of aliens. You destroy every one. But another set appears. These seem more intelligent. You eliminate them too. Your fuel supply is diminishing. You must destroy two more sets before you can dock. The space station is now on your scanner. With sound!

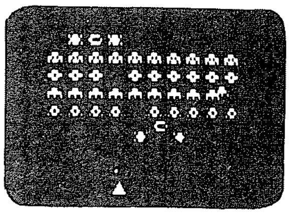
Price: $26.50



METEOR MISSION II

As you look down on your view, astronauts cry out for rescue. You must maneuver through the asteroids and meteors (Can you get back to the space station?) Fire lasers to destroy the asteroids, but watch out, there could be an alien Flagship lurking. Includes sound effects!

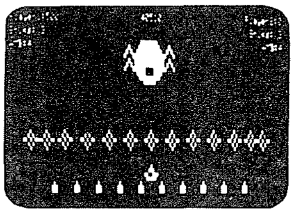
Price: $26.50



GALAXY INVASION

The sound of the klaxon is calling you! Invaders have been spotted warping toward Earth. You shift right and left as you fire your lasers. A few break formation and fly straight at you! You place your finger on the fire button knowing that this shot must connect! With sound effects!

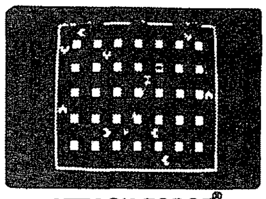
Price: $26.50



DEFENSE COMMAND

The invaders are back! Alone, you defend the all important nuclear fuel canisters from the repeated attacks of thieving aliens, repeatedly. An alien passes your guard, snatches a canister and flys straight off. Quick! You have one last chance to blast him from the sky! With sound and voice.

Price: $26.50



ATTACK FORCE®

As your ship appears on the bottom of the maze, eight alien ships appear on the top, all traveling directly at you! You move toward them and fire missiles. But the more aliens you destroy, the faster the remaining ones become. If you get too good you must endure the "Flagship". With sound effects!

Price: $26.50

FOR YOUR ENTERTAINMENT

MICRO-80 now offers you the widest range possible in entertainment software. These programs are supplied on cassette for the Level II/16K TRS-80 Model I/III (except as noted). They are also suitable for the System 80 but sound may not be available unless a hardware modification has been fitted to reverse the roles of recorders #1 and #2.

DEFENCE PENETRATOR $20.95
DEFENCE PENETRATOR is based on one of the most popular arcade favourites of all time with smooth graphics and sound effects. With realistic scrolling planetscape it's the best game yet.

BATTLE STATION $21.50
The aim of the game is to defend your space station against the attack of four alien space ships.

MORGOTH $20.95
Morgoth is a unique action packed adventure allowing you to wander through the enchanted dominion of Morgoth and collect the lost treasures of KAZARD KALLAHAN. But Beware! You must escape before the satanic Morgoth is aroused and seeks yea!

KILLER BEETLES $21.50
The aim of the game is to dig traps. When a beetle falls in you must fill it in to bury them, before they can catch you.

STAR CRESTA $20.95
Star Cresta takes you beyond the limits of your computer and into the Cosmic void itself! Beware! Iron clad concentration and lightning relfexes are required to destroy the evil empress.

JUNGLE RAIDERS $21.50
The aim of the game is to defend your four bases from the marauding Jungle Raiders. You kill all the Jungle Raiders and they try to hit you with their spears or drag off all four of your bases.

KILLER GORILLA $21.50
Four completely different frames. Each one offering a different challenge, makes this one of the most complex and stimulating games ever written for a TRS-80. The game keeps track of the top ten scores along with a six character name for each score.

JUNGLE BOY $21.50
The ultimate challenge! Are your reflexes fast enough to swing Jungle Boy from vine to vine? Can you swing through the jungle? Can you swim by the alligators? These are just some of the things you will find very challenging in Jungle Boy.

STELLAR WARP $20.95
Animation with superior fighter craft brings you an even greater challenge. As your computer advances your level, the aliens become more dangerous and the harder it is to stay alive!

HOPPY $21.50
The aim of the game is to get your frogs across the busy highway without being squashed and then across the river by means of floating logs and turtles.

INSECT FRENZY $21.50
The aim is to stop the centipede from getting you, all the time keeping an eye out for the giant spider.

ALIEN CRESTA $21.50
The aim is to defend your ship from numerous attacks from an assortment of aliens. If you get hit three times, it's all over.

RALLY RACER $20.95
Drive through an action packed maze and try to hit all the flags before Morgan the Mad motorist or Crazy Harry and his killer hoodlums catch you!

SPECIAL OFFER

Buy any two games on this page and pay only $35 SAVE $6

NOTE:
As the prices of imported software may vary, these prices are valid for current stock only and prices are subject to change without notice.

Double Your Disk Storage Capacity with the

LNDoubler 5/8

The LNDoubler is easily installed into your expansion interface and provides support for both 5¼" and 8" disk drives. Completely compatible with all the major Disk Operating Systems, the LNDoubler provides technically advanced, tested and reliable double-density operation with such features as:

- Analog phase lock loop data separation.
- Precision write precompensation.
- High quality PCB will all contacts gold-plated.
- Drives 1-3 may be software selected as 5" or 8" drives and a switch is provided for drive 0.
- Supports any mix of 5" or 8" drives, single or double density, single or double sided.

The LNDoubler will inrease the formatted storage capacity of **each** 40 track single-sided drive by 80% to over 180Kbytes — for just over half the cost of one disk drive. With an 8 inch double-sided double-density disk drive, you can have over one Megabyte of online storage!

The LNDoubler 5/8 doubler with documentation is available for ...

$285 plus $2.00 p.&p.

NOTE: A special cable is required for 8" drive operation and 8" double-density operation requires a 3.55 MHz CPU speed-up modification.

now available ...

DOSPLUS 3.5

is the state of the art in Disk Operating Systems for the Model 1 and Model 3 offering an order of magnitude increase in flexibility and performance over its predecessor DOSPLUS 3.4 and yet, is easier to use and more friendly with a Help facility explaining the syntax of DOS commands. The huge manual of over 350 pages describes the system in detail and is sectioned and tagged so that you can find what you want more quickly. Far greater flexibility is offered by the introduction of device drivers that are external to the system and that can be tailored to your needs.

Some of the features offered by DOSPLUS 3.5 are ...

- Single and double density support with density recognition.
- Improved file control facilities and date stamping of files.
- A keyboard driver that offers single key entry.
- An extensive Job Control Language.
- Complete and detailed technical system information.
- Two versions of BASIC, plus a BASIC label facility.
- and much more.

DOSPLUS 3.5 REPRESENTS EXCELLENT VALUE AT $160

When ordering by MAIL please specify Model 1 or Model 3 and include $2 for freight

DISK DRIVE HEADS CLEANING DISKETTES $44 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 has 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

5" FLOPPY DISKETTES 3M Scotch Brand

- Reinforced centre hole
- To suit soft sector format
- Rated for double density

Box of 10 single-sided for $59 incl p. & p.

Box of 10 double-sided for $69 incl p. & p.

FIT A PERCOM DATA SEPARATOR and eliminate CRC errors and locked-out tracks $45 plus $2 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.

HARDWARE KITS

Are you the do-it-yourself type? Then these kits are for you

PCG-80 Hiresolution Graphics Board

The ultimate hi-resolution graphics board for the TRS-80 Model 1 and System 80 that provides 256 programmable characters as well as the ability to emulate a 384 x 192 dot addressable screen. The hardware capabilities of the PCG-80 enable you to plot mathematical functions, display arcs, circles, lines, waves and design and display custom characters ranging from 72 pixels to 18432 pixels. You can mix graphics and characters anywhere on the screen.

PCG-80 to suit Model 1 only $199
plus $5 p. & p.
PCG-80 to suit System 80 only $219
plus $5 p. & p.

(Note: Not suitable for later type Japanese keyboards on TRS-80 Model 1).

Typewriter Interface Kit

The MICRO-80 interface does not interfere with the normal operation of your Praxis 35 or ET-121 typewriter which becomes a reliable, high qulaity, letter perfect printer. We are convinced that you will be able to fit the interface yourself without hassles and be printing within the hour. Our interfaces provide a Centronics compatible parallel interface and are supplied with power supply, comprehensive fitting instructions and user manual. The cable to connect the interface to the computer is not included in the price. We also offer a fitting service in Adelaide for only $35.

Inteface to suit Praxis 35 only $300
plus $10 p. & p.
Interface to suit ET-121 only $375
plus $10 p. & p.

16K Memory Upgrade Kit ONLY $25

plus $2.00 p. & p.

Large volume means we can buy better and can pass the savings on to you. There are our proven, prime, branded 200 nanosecond chips, guaranteed for 12 months. A pair of DIP shunts is also required to upgrade 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 install them.

Lower Case Modification $49

plus $2.00 p. & p.

The MICRO-80 modification features true below-the-line descenders, a block cursor and symbols for the 4 playing-card suits. Each kit comes with comprehensive fitting instructions and two universal lower-case driver routines on cassette to enable you to display lower case. These routines are self-relocating, self-protecting and will co-reside with other machine language programs (the second includes keyboard-debounce and flashing cursor). Fitting requires soldering inside the computer and should only be carried out by an experienced hobbyist or technician. A fitting service is available in capital cities for only $20.00 and a list of installers is included with each kit. (Specify TRS-80 Model I or System 80 when ordering.)

All prices include Sales Tax and are correct at time of publication but are subject to changes without notice. All equipment carries MICRO-80's Australia-wide 90-day warranty covering parts and labour.

DISK OPERATING SYSTEMS & DEVELOPMENT SOFTWARE

You can increase your programming productivity, the execution speed and 'user friendliness' of your programs by using an enhanced Disk Operating System (DOS). Together with the other utility software, you can get the most from your disk drives. Note: For DOSes, include $2.00 for freight.

DOSPLUS 3.5 **$160.00**
(Specify Model I or Model III)
DOSPLUS 3.5 is a powerful, sophisticated DOS intended for the experienced user. The system can be configured to suit your requirements, provides greatly enhanced features over 3.4 and new features like single-key entry, date-stamping of files, a Help file and more. More user friendly than 3.4, DOSPLUS 3.5 comes with a very extensive stand-alone manual.

BMON by Edwin Paay **$20.50**
The ultimate High Memory Basic Monitor L2/16-48K
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?

THE FLOPPY DOCTOR/MEMORY DIAGNOSTIC
Model III Disk **$43.50**
THE MICRO CLINIC offers two programs designed to thoroughly check out the two most trouble-prone sections of the TRS-80 – the disk system (controller and drives) and the memory arrays. Both programs are written in Z80 machine code and are supplied together on diskette for a minimum 32K, one disk system.

NEWDOS 80 VERSION 2.0 **$185.00**
(Specify Model I or Model III)
Newdos 80 suits the experienced user who has already used TRSDOS, understands the manual and is prepared to learn the somewhat complicated syntax of one of the most powerful DOS's available. With the correct hardware, Newdos 80 supports any mix of single- or double-sided, single or double density, 5" or 8" disk drives with track counts up to 96. It provides powerful, flexible file handling in BASIC including variable length records up to 4096 bytes. Definitely not for the beginner.

MASTER DISK DIRECTORY **$20.95**
FIND THE PROGRAM FAST!! PAYS FOR ITSELF BY RELEASING REDUNDANT DISK SPACE!! MASTER DIRECTORY records the directories of all your individual disks onto one directory disk. Then it allows you examine them, find an individual file quickly, list files alphabetically, weed out redundant files, identify disks with free space, list files by extension, etc., etc. This program is invaluable for the serious disk user and will pay for itself many times over. Not fully compatible with NEWDOS 80.

COLOUR COMPUTER SOFTWARE

For Tandy Colour Computers with at least 16K of memory, choose from the following selection:

SPACE FIGHTER **$20.95**
You are out in space, armed with laser beams and your mission is to shoot down five enemy fighters. Skill is required to keep your enemy in your sights to destroy them. Requires 16K Extended Colour BASIC.

ESCAPE FROM MARS **$20.95**
A good first adventure that takes place on Mars where you have to explore the city and deal with the possibility of hostile aliens.

BREAKAWAY **$20.95**
Written in machine language for speed and features 15 levels of difficulty.

QUEST **$25.50**
An adventure played on a map of Alesia generated by your computer. You must gather men and supplies by combat, exploration, trading, etc. When your force is strong enough, you attack the citadel in a battle to the finish.

RACING DRIVER **$23.50**
You drive along able to see only the road in front of you, leaving little time to avoid obstacles and hazards that you meet in your path. Featuring three levels of skill, colour and sound. Requires 16K Extended Colour BASIC.

HAUNTED HOUSE **$20.95**
A real adventure for the kids, with ghosts and goblins, treasures and problems.

TREK ADVENTURE **$20.95**
You are in deep trouble aboard a starship abandoned by its crew.

PYRAMID **$20.95**
An exciting and challenging adventure where you hunt for treasure in a pyrammid full of problems.

80 COLUMN DOT MATRIX PRINTER

Features:
- 80 cps bi-directional, logic seeking
- 40, 71, 80 or 142 characters per line
- Normal and italic alphanumeric, symbol and semi-graphic characters
- Unidirectional bit image graphics (8 x 640 or 8 x 1280 dot/line)
- Tractor and friction feed

EPSON MX80 compatible control codes

FOR A LOW $599

SPECIAL ANNOUNCEMENT

We are expanding our range of disk drives to include the new

SLIMLINE

MINI-FLOPPY DISK DRIVES

These half-height, 5¼" disk drives represent the state of the art in both technological design and mechanical construction. With the characteric high quality of manufacture expected in Japanese products, these drives feature ultra-modern electronics, servo-controlled direct drive motors and exceptional physical construction that provide extremely reliable, smooth and quiet operation in both single and double density. Two varieties will be available shortly: 40 track double sided and 80 track double sided.

Please WRITE for further information.

OTHER PRINTERS AVAILABLE:

EPSON RX-80	**$995**
Features: 100 cps, 6 character sizes, bit image and graphic modes.	
ITOH PROWRITER 8510	**$1150**
Features: 120 cps, bit graphics and proportional printing.	
EPSON FX-80	**$1399**
Features: 160 cps, 6 character sizes, proportional printing, bit graphics.	

All prices include Sales Tax and are correct at time of publication but are subject to change without notice.
All equipment carries MICRO-80's Australia-wide 90 day warranty covering parts and labour.
Add $10 road freight anywhere in Australia.

multiplies them, converts this value into a string value, and puts it in the array. Lines 330, 390 and 420 prevent inappropriate data entering the matrix. Line 330 also assigns a value in I(N) to each row of the string matrix. This array is used later when ordering the data.

Lines 450 to 570 echo the data (using lines 680 to 760) and allow lines to be added or changed, and are quite straightforward.

Perhaps the most important part of the program is found in lines 770 to 840 which arranges the values in descending order. Variable T sets the values to be compared in line 800, but it is not these values which are transposed in line 810. The values in the I(N) array are transposed instead. And the display subroutine of lines 680 to 760 prints the rows of data corresponding to the new order of I(N).

Lines 850 to 940 are a Screenprint subroutine. LN is used as a linecounter since the lineprinter counts only one line for a full scan of the screen. Lines 950 to 970 are a pagination subroutine, with the values set for A4 size pages.

Lines 980 to 1050 are NOT used in the usual running of the program. They are used only when developing and testing the program to avoid having to enter data by hand. They are called by line 320 which should be deleted when the program has been debugged.

- 0000000000 -

```
   **** TOWERS OF HANOI ****

         COLOUR COMPUTER


10 'THE TOWERS OF HANOI
20 '                BY
30 '            MICHAEL BYRNE
   MODIFIED FOR THE COLOR
   COMPUTER BY   MICRO-80
40 CLEAR 93
50 CLS(0):GOSUB 1430:GOSUB 1510:
MAX=8
60 DIM N(3),P1(MAX),P2(MAX),P3(M
AX),SK(10*MAX)
70 SC=1:AL=2:DN=3:CT=0:P1(0)=99:
P2(0)=99:P3(0)=99
80 NM=0:J=0:I=0:SP=0:B=0:CLS
90  INPUT"HOW MANY DISCS DO YOU
WANT";CT
100 IF CT> MAX THEN PRINT"MAXIMU
M NUMBER IS 8":GOTO90
110  IF CT<=0 THEN PRINT"SORRY,
THAT'S NOT POSSIBLE":GOTO 90
120  IF CT<=3 THEN PRINT"YOU HAVE
 NO SENSE OF ADVENTURE  BUT I SU
PPOSE YOU HAVE TO START SOMEWHER
E"
130 N(1)=CT:N(2)=0:N(3)=0
140 INPUT"AUTOMATIC OR MANUAL (A
 OR M)";MODE$
150 IF MODE$<>"A" GOTO 270
160 PRINT"HOW FAST":INPUT" (0 IS
 FASTEST, 10 IS SLOWEST)";S
170 CLS:GOSUB1150
180 GOSUB530:SP=4
190 SK(SP-3)=CT:SK(SP-2)=SC:SKK(
SP-1)=DN:SK(SP)=AL
200 B=SP:SP=SP+4
210 SK(SP-3)=SK(B-3):SK(SP-2)=SK
(B-2):SK(SP-1)=SK(B-1):SK(SP)=SK
(B)
220 IF CT< >1 THEN GOTO 250
230 SP=SP+2:SK(SP-1)=1:SK(SP)=3:
GOSUB 580
240 GOTO260
250 GOSUB 810
260 PRINT@416,"";:GOTO500
270 CLS:GOSUB 1150:S=0
280 GOSUB 530
290 PRINT@416,"
                  ";
300 PRINT@416,"FROM PEG";:INPUT
F
310 PRINT@430,"TO PEG";:INPUT T
320 IF F<1 OR T<1 OR F>3 OR T>3
THEN PRINT"NO SUCH PEG":GOTO 290
330 IF F=T THEN PRINT"NOT ALLOWE
D":GOTO 290
340 IF N(F)<=0 THEN PRINT"THERE
ARE NO DISCS ON PEG ";F:GOTO290
350 I=N(F):J=N(T)
360 ON F GOTO 370,400,420
370 T1=P1(I)
380 IF T=2 THEN T2=P2(J):ELSE T2
=P3(J)
390 GOTO 430
400 T1=P2(I):IF T=1 THEN T2=P1(J
):ELSE T2=P3(J)
410 GOTO 430
420 T1=P3(I):IF T=2 THEN T2=P2(J
):ELSE T2=P1(J)
430 IF T1>T2 THEN PRINT"NOT ALLO
WED":GOTO290
440 SP=SP+2:SK(SP-1)=F:SK(SP)=T
450 GOSUB 580
460 IF N(1)=0 AND N(2)=0 AND N(3
)=CT THEN GOTO 470:ELSE GOTO 290
470 M=2^CT-1:PRINT@416,"
                        ":PRINT@
416,"";
480 IF NM=M THEN PRINT"CONGRATUL
ATIONS! YOU DID IT" ELSE IF NM-M
<=M*0.15 THEN PRINT"NOT BAD AT A
LL"
490 PRINT"YOU TRANSFERRED ALL TH
E DISCS IN ";NM;" MOVES
500 PRINT"CARE FOR ANOTHER GAME"
:PRINT"(YES OR NO)";
510 INPUT AN$
520 IF AN$="N" OR AN$="NO"THEN C
LS:POKE359,126:END ELSE GOTO80
530 J=CT
540 FOR I=1 TO CT
550 P1(I)=J:J=J-1:NEXT I
560 RETURN
570 'SHIFT DISCS
580 NM=NM+1:SE=SK(SP-1):DE=SK(SP
)
590 FOR I=1 TO 30*S:NEXT I
600 I=N(SE):Y=23-(I*2)
610 IF SE< >1 THEN 630
620 TD=P1(I):X=10:GOTO 660
```

```
630 IF SE< >2 THEN 650
640 TD=P2(I):X=32:GOTO 660
650 TD=P3(I):X=52
660 P=2:SIZE=TD
670 GOSUB 1320
680 I=N(DE)+1:Y=23-(I*2)
690 IF DE< >1 THEN 710
700 P1(I)=TD:X=10:GOTO 740
710 IF DE< >2 THEN 730
720 P2(I)=TD:X=32:GOTO 740
730 P3(I)=TD:X=52
740 P=1
750 GOSUB 1320
760 N(SE)=N(SE)-1
770 N(DE)=N(DE)+1
780 SP=SP-2
790 PRINT@20,"MOVE ";NM;
800 RETURN
810 'HANOI(RECURSIVE PROCEDURE)
820 IF SK(SP-3)>2 THEN GOTO 830
ELSE GOTO 1000
830 B=SP:SP=SP+4
840 SK(SP-3)=SK(B-3)-1
850 SK(SP-2)=SK(B-2)
860 SK(SP-1)=SK(B)
870 SK(SP)=SK(B-1)
880 GOSUB 810
890 B=SP:SP=SP+2
900 SK(SP-1)=SK(B-2)
910 SK(SP)=SK(B-1)
920 GOSUB 580
930 B=SP:SP=SP+4
940 SK(SP-3)=SK(B-3)-1
950 SK(SP-2)=SK(B)
960 SK(SP-1)=SK(B-1)
970 SK(SP)=SK(B-2)
980 GOSUB 810
990 GOTO 1120
1000 B=SP:SP=SP+2
1010 SK(SP-1)=SK(B-2)
1020 SK(SP)=SK(B)
1030 GOSUB 580
1040 B=SP:SP=SP+2
1050 SK(SP-1)=SK(B-2)
1060 SK(SP)=SK(B-1)
1070 GOSUB 580
1080 B=SP:SP=SP+2
1090 SK(SP-1)=SK(B)
1100 SK(SP)=SK(B-1)
1110 GOSUB 580
1120 SP=SP-4
1130 RETURN
1140 'DRAWPEGS
1150 CLS(0):REM ' PROC DRAWPEGS
1160 FOR I=1 TO 8
1170 POKE1093+I*32,191:POKE1104+
I*32,191:POKE1114+I*32,191
1180 NEXT I
1190 FORI=1376TO1407:POKE I,191:
NEXTI
1200 PRINT:PRINT"      1
 2          3"
1210 Y=21:X=10
1220 FOR I=CT TO 1 STEP -1
1230    FOR J=1 TO I
1240 SOUND 10*I,1
1250 SET(X-J,Y,I):SET(X-J-1,Y,I)
:SET(X-J,Y-1,I):SET(X-J-1,Y-1,I)
1260 SET(X+J+2,Y,I):SET(X+J+1,Y,
I):SET(X+J+1,Y-1,I):SET(X+J+2,Y-
1,I)
1270 NEXT J
1280 Y=Y-2
1290 NEXT I
1300 RETURN
1310 'SWITCH PROCEDURE
1320 IF P=2 THEN 1390
1330 FOR J=1 TO SIZE
1340 C=SIZE
1350 SOUND 10*C,1
1360 SET(X-J-1,Y,C):SET(X-J-1,Y-
1,C):SET(X+J+1,Y,C):SET(X+J+1,Y-
1,C):SET(X-J,Y-1,C):SET(X+J+2,Y-
1,C):SET(X-J,Y,C):SET(X+J+2,Y,C)
1370 NEXT J
1380 GOTO 1420
1390 FOR J=1 TO SIZE
1400 RESET(X-J-1,Y):RESET(X-J-1,
Y-1):RESET(X+J+1,Y):RESET(X+J+1,
Y-1):RESET(X-J,Y-1):RESET(X+J+2,
Y-1):RESET(X-J,Y):RESET(X+J+2,Y)
1410 NEXT J
1420 RETURN
1430 D$= STRING$(32,"*")
1440 PRINT@128,D$
1450 PRINT D$:PRINT
1460  PRINT@231,"T H E    T O W
E R S"
1470 PRINT@265,"O F    H A N O I
":PRINT
1480  PRINT D$
1490 PRINT D$
1500 RETURN
1510 POKE359,57:SCREEN0,1:GOSUB1
900: FORI=1 TO 1000:NEXT I
1520 PRINT@449,"INSTRUCTIONS (YE
S OR NO)";:INPUT AN$
1530 IF AN$< >"Y" AND AN$< >"YES
" THEN RETURN
1540  CLS
1550  PRINT@10,"INSTRUCTIONS"
1560 PRINT@42,"------------":PRI
NT
1570  PRINT"THERE ARE THREE PEGS
.  ON ONE OFTHESE IS ARRANGED, I
N  ORDER  OFDECREASING SIZE, A N
UMBER OF     DISCS. THE OBJECT IS
 TO MOVE ALLTHE DISCS FROM THE L
EFT MOST PEG (1) TO THE RIGHT MO
ST PEG (3)."
1580 PRINT"SUBJECT TO THE FOLLOW
ING"
1590 PRINT"CONSTRAINTS :-":PRINT
1600 GOSUB1870:CLS
1610  PRINT"1. ONLY 1 DISC MAY B
E MOVED AT      A TIME"
1620 PRINT"2. A DISC MAY NOT BE
PLACED ON      TOP OF A DISC WHIC
H IS           SMALLER"
1630 GOSUB 1870
1640 PRINT@10,"PLAYING HANOI"
1650 PRINT@42,"-------------":PR
INT
1660  PRINT"THERE ARE TWO MODES
OF PLAY.     THESE ARE <A>UTOMATI
C AND        <M>ANUAL AND YOU WIL
L BE ASKED  TO SELECT ONE.""
1670  PRINT"AUTOMATIC OPERATION"
1680 PRINT"THE COMPUTER WILL ASK
 YOU HOW   MANY DISCS YOU WOULD
LIKE MOVED AND HOW FAST YOU WOUL
D LIKE THEMMOVED. IT WILL THEN P
ROCEED TO  TRANSFER THE DISCS FR
OM PEG TO  PEG."
1690 GOSUB 1870
1700 PRINT@10,"PLAYING HANOI"
1710 PRINT@42,"-------------":PR
INT
1720  PRINT"MANUAL OPERATION"
1730 PRINT"HERE YOU WILL BE ASKE
D HOW MANY DISCS YOU WOULD LIKE
TO MOVE.  THEN YOU WILL BE PROM
PTED TO   ";
1740 PRINT" TYPE THE NUMBER OF T
HE PEG THE  DISC IS TO BE TAKEN
OFF AND THE NUMBER THE PEG THE D
ISC IS TO BEPLACED ON."
1750 PRINT"WHEN YOU HAVE SUCCESS
FULLY TRAN-SFERRED ALL DISCS YOU
 WILL BE   TOLD HOW MANY MOVES Y
OU TOOK"
1760 GOSUB1870
1770 PRINT@10,"SUGGESTION"
1780 PRINT@42,"----------":PRINT
1790  PRINT"IF YOU ARE UNSURE OF
 THE GAME   TRY AUTOMATIC OPERAT
ION WITH    ABOUT FOUR DISCS AND
 LOW SPEED  (E.G. 10). THEN SIT
BACK AND LETTHE COMPUTER DO ALL
THE WORK"
```

```
1800 PRINT"AFTER ALL THAT IS WHA
T IT IS    THERE FOR.":PRINT:GOS
UB1870:CLS
1810  PRINT"FOR A GIVEN NUMBER O
F DISCS, N, THE SMALLEST NUMBER
OF MOVES    REQUIRED IS GIVEN BY
 :-"
1820 PRINT"M = 2^N-1"
1830 PRINT"SO FOR 3 DISCS THAT'S
 7 MOVES,  4 DISCS 15 MOVES UP T
O THE MAX' OF 8 DISCS WHICH REQU
IRES 255   MOVES.":PRINT
1840  PRINT"    HAVE FUN!"
1850 GOSUB 1870
1860 RETURN
1870  PRINT@480,"PRESS <ENTER> W
HEN READY";
1880 INPUT D$:CLS
1890  RETURN
1900 FORX=1024TO1055:POKEX,191:P
OKEX+480,191:NEXTX:FORY=0TO16:PO
KE1024+Y*32,191:POKE1055+Y*32,19
1:NEXTY
1940 RETURN
```

**** OTHELLO ****

COLOUR COMPUTER

```
10 '    ********************
        *  (C)  S.GIBBONS  *
        * 34 THE COMENARRA *
        *PARKWAY,THORNLEIGH*
        *    N.S.W. 2120   *
        ********************
20 CLS
30 PRINT"              OTHELLO":P
RINT:PRINT:PRINT
40 FORE=255TO1STEP-20:SOUNDE,1:N
EXT
50 PRINT"DO YOU NEED INSTRUCTION
S??":FORE=255TO1STEP-20:SOUNDE,1
:NEXT
60 I$=INKEY$:IFI$="Y"THENGOTO126
0ELSEIFI$="N"THEN80
70 GOTO60
80 CLS0
90 GH=2
100 POKE1036,15:POKE1037,20:POKE
1038,8:POKE1039,5:POKE1040,12:PO
KE1041,12:POKE1042,15
 110 Y=6
120 FORX=22TO41:SET(X,7,3):SET(X
,6,3):SET(X,24,3):SET(X,25,3):SE
T(23,Y,3):SET(22,Y,3):SET(40,Y,3
):SET(41,Y,3):Y=Y+1:NEXT
130 FORX=24TO39:FORY=8TO23:SET(X
,Y,6):NEXT
140 NEXT
150 Z=49
160 FORX=1131+32TO1131+64+192STE
P32
170 POKEX-1,Z:POKEX+10,Z:Z=Z+1:N
EXT
180 Z=49
190 FOR X=1090+10 TO1097+10:POKE
X,Z:POKEX+352,Z:Z=Z+1:NEXT
200 CL$=CHR$(128)+CHR$(128)+CHR$
(128)+CHR$(128)+CHR$(128)+CHR$(1
28)+CHR$(128)+CHR$(128):CL$=CL$+
CL$+CL$+CL$
210 SET(30,15,5):RESET(32,14):RE
SET(33,14):RESET(32,15):RESET(33
,15)
220 RESET(30,16):RESET(31,16):RE
SET(30,17):RESET(31,17):SET(32,1
7,5)
230 PRINT@448,CL$;
240 IF GH=1 THENPRINT@448,"";:IN
PUT"WHITE'S TURN";A,B:C=5:OC=0
250 PRINT@448,CL$;
260 IF GH=2 THENPRINT@448,"";:IN
PUT"BLACK'S TURN";A,B:C=0:OC=5
270 IFA<1 ORA>8 OR B<1 OR B>8 TH
ENPRINT@448,"INVALID MOVE";:FORE
=1TO1000:NEXT:GOTO230
280 GH=GH+1:IFGH>2THENGH=1
290 X=2*(A+12)-1:Y=2*(B+3)
300 IF POINT(X,Y)=5 OR POINT(X,Y
)=0 THENPRINT@448,"INVALID MOVE"
;:FORE=1TO1000:NEXT:IF C=0 THEN
GH=2:GOTO230ELSEGH=1:GOTO230
310 IF POINT(X+2,Y+2)=5 OR POINT
(X+2,Y+2)=0 OR POINT(X-2,Y+2)=0
OR POINT(X-2,Y+2)=5 OR POINT(X-2
,Y-2)=0 OR POINT(X-2,Y-2)=5 THEN
ABC=1ELSEABC=0
320 IF ABC=1 OR POINT(X+2,Y-2)=5
 OR POINT(X+2,Y-2)=0 OR POINT(X-
2,Y)=0 OR POINT(X-2,Y)=5 OR POIN
T(X+2,Y)=0 OR POINT(X+2,Y)=5 OR
POINT(X,Y-2)=5 OR POINT(X,Y-2)=0
 OR POINT(X,Y+2)=0 OR POINT(X,Y+
2)=5 THEN780
330 PRINT@448,"INVALID MOVE";:FO
RE=1TO1000:NEXT:IFGH=2THENGH=1:G
OTO230ELSEGH=2:GOTO230
340 GOTO230
350 IFC=5 THEN SET(X,Y,5)ELSE RE
SET(X,Y):RESET(X-1,Y):RESET(X,Y+
1):RESET(X-1,Y+1)
360 IF POINT(X,Y-2)=OC THEN370EL
SE440
370 M=Y
380 M=M-2:IFM<6 OR POINT(X,M)=6
THEN440
390 IF POINT(X,M)=C THEN400 ELSE
 380
400 M=Y
410 M=M-2:SOUND252,1:IF POINT(X,
M)=C THEN440
420 IF C=5 THENGOSUB790ELSEGOSUB
800:GOTO410
430 GOTO410
440 IF POINT(X,Y+2)=OC THEN450EL
SE520
450 M=Y
460 M=M+2:IFM>23 OR POINT(X,M)=6
 THEN520
470 IF POINT(X,M)=C THEN480ELSE4
60
480 M=Y
490 M=M+2:SOUND251,1:IF POINT(X,
M)=C THEN520
500 IFC=5 THENGOSUB790ELSEGOSUB8
00:GOTO490
510 GOTO490
520 IF POINT(X-2,Y)=OC THEN530EL
SE600
530 N=X
540 N=N-2:IFN<23 OR POINT(N,Y)=6
 THEN600
550 IF POINT(N,Y)=C THEN560ELSE5
40
560 N=X
570 N=N-2:SOUND253,1:IFPOINT(N,Y
)=C THEN600
580 IFC=5THENSET(N,Y,5):SET(N-1,
Y,5):SET(N,Y+1,5):SET(N-1,Y+1,5)
ELSERESET(N,Y):RESET(N-1,Y):RESE
T(N,Y+1):RESET(N-1,Y+1):GOTO570
590 GOTO570
600 IFPOINT(X+2,Y)=OC THEN610 EL
SE680
610 N=X
620 N=N+1:IFN>40 OR POINT(N,Y)=6
 THEN680
630 IF POINT(N,Y)=C THEN640ELSE6
20
640 N=X
650 N=N+2:SOUND254,1:IFPOINT(N,Y
)=C THEN680
```

```
660 IF C=5 THENSET(N,Y,5):SET(N-
1,Y,5):SET(N,Y+1,5):SET(N-1,Y+1,
5)ELSERESET(N,Y):RESET(N-1,Y):RE
SET(N,Y+1):RESET(N-1,Y+1)
670 GOTO650
680 IF POINT(X+2,Y-2)=OC THEN 69
0ELSE760
690 N=X:M=Y
700 N=N+2:M=M-2:IF M<6 OR N>40 O
R POINT(N,M)=6 THEN 760
710 IFPOINT(N,M)=C THEN720ELSE70
0
720 N=X:M=Y
730 N=N+2:M=M-2:SOUND255,1:IFPOI
NT(N,M)=C THEN760
740 IF C=5 THENSET(N,M,5):SET(N-
1,M,5):SET(N,M+1,5):SET(N-1,M+1,
5)ELSERESET(N,M):RESET(N-1,M):RE
SET(N,M+1):RESET(N-1,M+1)
750 GOTO730
760 GOSUB930
770 GOTO830
780 GOTO350
790 SET(X,M,5):SET(X-1,M,5):SET(
X,M+1,5):SET(X-1,M+1,5):RETURN
800 RESET(X,M):RESET(X-1,M):RESE
T(X,M+1):RESET(X-1,M+1):RETURN
810 SET(N,M,5):SET(N-1,M,5):SET(
N,M+1,5):SET(N-1,M+1,5):RETURN
820 RESET(N,M):RESET(N-1,M):RESE
T(N,M+1):RESET(N-1,M+1):RETURN
830 WHS=0:BLS=0:SP=0
840 FORX=22TO40STEP2:FORY=8TO24S
TEP2
850 IFPOINT(X,Y)=0 THENBLS=BLS+1
860 IF POINT(X,Y)=6 THENSP=SP+1
870 IFPOINT(X,Y)=5 THENWHS=WHS+1
880 NEXT:NEXT
890 PRINT@32+21,"WHITE:";WHS;
900 PRINT@64+21,"BLACK:";BLS;
910 IF SP=0 THEN1190
920 GOTO230
930 IF POINT(X-2,Y-2)=OC THEN940
ELSE1010
940 N=X:M=Y
950 M=M-2:N=N-2:IFM<6 OR N<23 OR
 POINT(N,M)=2 THEN1010
960 IF POINT(N,M)=C THEN970ELSE9
50
970 N=X:M=Y
980 M=M-2:N=N-2:SOUND249,1:IFPOI
NT(N,M)=C THEN1010
990 IFC=5 THENGOSUB810 ELSE GOSU
B820
1000 GOTO980
```

```
1010 IF POINT(X+2,Y+2)=OC THEN10
20ELSE1090
1020 N=X:M=Y
1030 M=M+2:N=N+2:IF M>23 OR N>40
 OR POINT(N,M)=6 THEN1090
1040 IF POINT(N,M)=C THEN1050ELS
E1030
1050 N=X:M=Y
1060 M=M+2:N=N+2:SOUND248,1:IFPO
INT(N,M)=C THEN1090
1070 IF C=5 THENGOSUB810ELSEGOSU
B820
1080 GOTO1060
1090 IF POINT(X-2,Y+2)=OC THEN11
00ELSE1170
1100 N=X:M=Y
1110 N=N-2:M=M+2:IFN<21 OR M>23
OR POINT(N,M)=6 THEN1170
1120 IF POINT(N,M)=C THEN1130ELS
E1110
1130 N=X:M=Y
1140 M=M+2:N=N-2:SOUND247,1:IFPO
INT(N,M)=C THEN1170
1150 IF C=5 THENGOSUB810ELSEGOSU
B820
1160 GOTO1140
1170 '
1180 RETURN
1190 PRINT@448,"";
1200 IFBLS>WHS THENPRINT"BLACK W
INS BY";BLS-WHS;"POINTS.";
1210 PRINT"WHITE WINS BY";WHS-BL
S;"POINTS.";
1220 FORE=1TO4000:NEXT:PRINT@448
,CL$;
1230 PRINT@448,"ANOTHER GAME??";
1240 I$=INKEY$:IFI$="Y"THENRUN
1250 IFI$="N"THENENDELSE1240
1260 CLS
1270 PRINT"                OTHELLO"
1280 GOSUB1640
1290 PRINT:PRINT"OTHELLO IS A GA
ME OF STRATEGY   FOR TWO PLAYERS
. IT CONSISTS OF 64 DOUBLE SIDED
 DISCS (IN THIS  CASE SQUARES) W
HICH ARE BLACK   ON ONE SIDE AND
 WHITE ON THE    OTHER."
1300 GOSUB1640
1310 PRINT"EACH PLAYER CHOOSES H
IS COLOUR  BLACK OR WHITE WHICH
IS HIS     COLOUR THROUGHOUT THE
 GAME.     THE GAME STARTS WITH
FOUR DISCS  IN THE MIDDLE, TWO OF
 EACH      EACH COLOUR CROSSING
EACH OTHER."
```

```
1320 GOSUB1640
1330 PRINT@448,"HIT <SPACEBAR> T
O CONTINUE."
1340 GOSUB1640
1350 I$=INKEY$:IFI$=" "THEN1360E
LSE1350
1360 GOSUB1650:CLS
1370 PRINT"                OTHELLO"
:PRINT
1380 GOSUB1640
1390 PRINT"BLACK ALWAYS STARTS F
IRST."
1400 PRINT"A MOVE CONSISTS OF 'O
UTFLANKING'YOUR OPPONENTS DISC(S
) THEN     FLIPPING THE OUTFLANK
ED DISC(S) OVER TO YOUR COLOUR (
WHICH THE  COMPUTER DOES FOR YOU
)."
1410 GOSUB1640
1420 PRINT
1430 PRINT"TO PLACE A DISC GIVE
THE 'X'    COORDINATE FOLLOWED B
Y A COMMA  ',' THEN THE 'Y' COOR
DINATE."
1440 GOSUB1640
1450 PRINT@448,"HIT <SPACEBAR> T
O CONTINUE"
1460 GOSUB1640
1470 I$=INKEY$:IFI$=" "THEN1480E
LSE1470
1480 GOSUB1650:CLS:PRINT"
       OTHELLO":PRINT
1490 GOSUB1640
1500 PRINT"TO OUTFLANK MEANS TO
PLACE A    DISC SO THAT YOUR OPP
ONENTS ROW OF DISC(S) IS BORDERE
D AT EACH  END BY A DISC OF YOUR
 COLOUR."
1510 GOSUB1640
1520 PRINT"A DISC MAY OUTFLANK I
N ANY      DIRECTION: HORAZONTAL
LY,        VERTICALLY OR DIAGONA
LLY."
1530 GOSUB1640
1540 PRINT"A DISC MAY OUTFLANK I
N ANY      NUMBER OF DIRECTOINS
AT THE SAMETIME. A DISC CAN ONLY
 BE PLACED ONE SQUARE AWAY FROM
ANY DISC INANY DIRECTION FROM TH
E DISC."
1550 GOSUB1640
1560 D$="............HIT <SPACE
BAR> FOR GAME OR <ENTER> FOR INS
TRUCTIONS AGAIN.................
."
```

```
1570 FORZ=1TOLEN(D$)
1580 I$=INKEY$
1590 PRINT@448+10,MID$(D$,Z,11)
1600 IFI$=" "THENGOSUB1650:CLS:G
OTO80
1610 IFI$=CHR$(13)THENGOSUB1650:
CLS:GOTO1260
1620 FORE=1TO100:NEXT
1630 NEXT:GOTO1570
1640 FORE=255TO1STEP-20:SOUNDE,1
:NEXT:RETURN
1650 FORE=1TO255STEP20:SOUNDE,1:
PRINT:NEXT:RETURN
```

**** TOWERS OF HANOI ****

HITACHI PEACH

```
10 '    ====  THE TOWERS OF HANOI  ====
20 '                BY
30 '           MICHAEL BYRNE
40 '           P.O. BOX 1826
50 '           CANBERRA CITY
60 '           A.C.T.   2601
70 CLEAR 93
80 CLS:GOSUB 1450:GOSUB 1530:MAX=10:SCRE
EN0,,1
90 DIM N(3),P1(MAX),P2(MAX),P3(MAX),SK(1
0*MAX)
100 SC=1:AL=2:DN=3:CT=0:P1(0)=99:P2(0)=9
9:P3(0)=99
110 NM=0:J=0:I=0:SP=0:B=0:CLS
120 INPUT"HOW MANY DISCS WOULD YOU LIKE"
;CT
130 IF CT>MAX THEN PRINT"THAT WILL TAKE
SOME TIME AND FIRST YOU WILLHAVE TO  CHA
NGE LINE 10":END
140 IF CT<=0 THEN PRINT"SORRY, THAT'S NO
T POSSIBLE":GOTO 120
150 IF CT<=3THEN PRINT"YOU HAVE NO SENSE
 OF ADVENTURE - BUT I SUPPOSEYOU HAVE TO
 START SOMEWHERE"
160 N(1)=CT:N(2)=0:N(3)=0
170 PRINT:PRINT"AUTOMATIC OR MANUAL  (A O
R M)";:GOSUB2050
180 IF IN$<>"A" GOTO 300
190 PRINT:INPUT"HOW FAST  (0 IS FASTEST,
 10 IS SLOWEST)";S
200 CLS:GOSUB1180
210 GOSUB560:SP=4
220 SK(SP-3)=CT:SK(SP-2)=SC:SKK(SP-1)=DN
:SK(SP)=AL
230 B=SP:SP=SP+4
240 SK(SP-3)=SK(B-3):SK(SP-2)=SK(B-2):SK
(SP-1)=SK(B-1):SK(SP)=SK(B)
250 IF CT<>1 THEN GOTO 280
260 SP=SP+2:SK(SP-1)=1:SK(SP)=3:GOSUB 61
0
270 GOTO290
280 GOSUB 840
290 LOCATE7,16:PRINT"";:GOTO530
300 CLS:GOSUB 1180:S=0
310 GOSUB 560
320 LOCATE7,18:PRINT"
                                  ";
330 LOCATE7,18:PRINT"FROM PEG";:INPUT F
340 LOCATE35,18:PRINT"TO PEG";:INPUT T:L
OCATE0,19:PRINT"
       ":LOCATE0,19
350 IF F<1 OR T<1 OR F>3 OR T>3 THENPRIN
T"NO SUCH PEG                       ":GOTO
 320
360 IF F=T THENPRINT"NOT ALLOWED
          ":GOTO 320
370 IF N(F)<=0THENPRINT"THERE ARE NO DIS
CS ON PEG ";F:GOTO320
380 I=N(F):J=N(T)
390 ON F GOTO 400,430,450
400 T1=P1(I)
410 IF T=2 THEN T2=P2(J):ELSE T2=P3(J)
420 GOTO 460
430 T1=P2(I):IF T=1 THEN T2=P1(J):ELSE T
2=P3(J)
440 GOTO 460
450 T1=P3(I):IF T=2 THEN T2=P2(J):ELSE T
2=P1(J)
460 IF T1>T2 THENPRINT"NOT ALLOWED
               ":GOTO320
470 SP=SP+2:SK(SP-1)=F:SK(SP)=T
480 GOSUB 610
490 IF N(1)=0 AND N(2)=0 AND N(3)=CT THE
N GOTO 500:ELSE GOTO 320
500 M=2^CT-1:LOCATE7,18:PRINT"
                                  ":LOCAT
E7,18:PRINT"";
510 LOCATE1,19:IF NM=M THENPRINT"CONGRAT
ULATIONS! YOU DID IT" ELSE IF NM-M<=M*0.
15 THENPRINT"NOT BAD AT ALL"
520 LOCATE1,20:PRINT"YOU TRANSFERRED ALL
 THE DISCS IN ";NM;" MOVES
530 LOCATE1,21:PRINT"CARE FOR ANOTHER GA
ME (YES OR NO)";
540 GOSUB2050:AN$=IN$
550 IF AN$="N" THEN CLS:END ELSE GOTO110
560 J=CT
570 FOR I=1 TO CT
580 P1(I)=J:J=J-1:NEXT I
590 RETURN
600 '============  SHIFT DISCS
610 NM=NM+1:SE=SK(SP-1):DE=SK(SP)
620 FOR I=1 TO 30*S:NEXT I
630 I=N(SE):Y=16-(I-1)
640 IF SE<>1 THEN 660
650 TD=P1(I):X=17:GOTO 690
660 IF SE<>2 THEN 680
670 TD=P2(I):X=40:GOTO 690
680 TD=P3(I):X=65
690 P=2:SIZE=TD
700 GOSUB 1360
710 I=N(DE)+1:Y=16-(I-1)
720 IF DE<>1 THEN 740
730 P1(I)=TD:X=17:GOTO 770
740 IF DE<>2 THEN 760
750 P2(I)=TD:X=40:GOTO 770
760 P3(I)=TD:X=65
770 P=1
780 GOSUB 1360
790 N(SE)=N(SE)-1
800 N(DE)=N(DE)+1
810 SP=SP-2
820 LOCATE30,3:PRINT"MOVE ";NM
830 RETURN
840 '============  HANOI(RECURSIVE PROC
EDURE)
850 IF SK(SP-3)>2 THEN GOTO 860 ELSE GOT
O 1030
860 B=SP:SP=SP+4
870 SK(SP-3)=SK(B-3)-1
880 SK(SP-2)=SK(B-2)
890 SK(SP-1)=SK(B)
900 SK(SP)=SK(B-1)
910 GOSUB 840
920 B=SP:SP=SP+2
930 SK(SP-1)=SK(B-2)
940 SK(SP)=SK(B-1)
950 GOSUB 610
960 B=SP:SP=SP+4
970 SK(SP-3)=SK(B-3)-1
980 SK(SP-2)=SK(B)
990 SK(SP-1)=SK(B-1)
1000 SK(SP)=SK(B-2)
1010 GOSUB 840
1020 GOTO 1150
1030 B=SP:SP=SP+2
1040 SK(SP-1)=SK(B-2)
1050 SK(SP)=SK(B)
1060 GOSUB 610
1070 B=SP:SP=SP+2
1080 SK(SP-1)=SK(B-2)
1090 SK(SP)=SK(B-1)
1100 GOSUB 610
1110 B=SP:SP=SP+2
1120 SK(SP-1)=SK(B)
1130 SK(SP)=SK(B-1)
```

```
1140 GOSUB 610
1150 SP=SP-4
1160 RETURN
1170 '============  DRAWPEGS
1180 CLS:' PROC DRAWPEGS
1190 LOCATE7,6:PRINT"";:A$=CHR$(142)+CHR
$(142)+CHR$(142)
1200 FOR I=1 TO 10
1210 LOCATE7,6+I: PRINT"             ";A$;"
                             ";A$;"
       ";A$
1220 NEXT I
1230 PRINT"      ";
1240 FOR I=1 TO 25:PRINT A$;:NEXT I
1250 PRINT:LOCATE7,4 :PRINT"            1
                           2
      3"
1260 Y=16:X=17
1270 FOR I=CT TO 1 STEP -1
1280    FOR J=1 TO I
1290 LOCATEX-J,Y:PRINTCHR$(142);
1300 LOCATEX+J+2,Y:PRINTCHR$(142);
1310    NEXT J
1320    Y=Y-1
1330 NEXT I
1340 RETURN
1350 '============  SWITCH PROCEDURE
1360 IF P=2 THEN 1410
1370 FOR J=1 TO SIZE
1380 LOCATEX-J,Y:PRINTCHR$(142);:LOCATEX
+J+2,Y:PRINTCHR$(142);
1390 NEXT J
1400 GOTO 1440
1410 FOR J=1 TO SIZE
1420 LOCATEX-J,Y:PRINTCHR$(32);:LOCATEX+
J+2,Y:PRINTCHR$(32);
1430 NEXT J
1440 RETURN
1450 D$=STRING$(63,"*")
1460 LOCATE7,7:PRINTD$
1470 LOCATE7,8:  PRINT D$:PRINT
1480 LOCATE25,10:PRINT"T H E    T O W E
R S"
1490 LOCATE27,11:PRINT"O F    H A N O I"
:PRINT
1500 LOCATE7,13: PRINT D$
1510 LOCATE7,14:PRINT D$
1520 RETURN
1530 FORI=1 TO 1000:NEXT I
1540 LOCATE7,17:PRINT"DO YOU WANT INSTRU
CTIONS (YES OR NO)";:GOSUB2050:AN$=IN$
1550 IF AN$<>"Y" THEN RETURN
1560 CLS
1570 LOCATE30,3:PRINT"INSTRUCTIONS"
1580 LOCATE30,4:PRINT"------------":PRIN
T
1590 PRINT"    THERE ARE THREE PEGS.  ON
 ONE OF THESE IS ARRANGED,
1600 PRINT"IN ORDER OF DECREASING SIZE,
A NUMBER OF DISCS.":PRINT
1610 PRINT"    THE OBJECT IS TO MOVE ALL
 THE DISCS FROM THE LEFTMOST"
1620 PRINT"PEG (1) TO THE RIGHTMOST PEG
(3), SUBJECT TO THE FOLLOWING"
1630 PRINT"CONSTRAINTS :-":PRINT
1640 PRINT"    1. ONLY ONE DISC MAY BE M
OVED AT A TIME"
1650 PRINT"    2. A DISC MAY NOT BE PLAC
ED ON TOP OF A DISC WHICH"
1660 PRINT"IS SMALLER"
1670 GOSUB 2020
1680 LOCATE30,3:PRINT"PLAYING HANOI"
1690 LOCATE30,4:PRINT"-------------":PRI
NT
1700 PRINT"    THERE ARE TWO MODES OF PL
AY.  THESE ARE <A>UTOMATIC"
1710 PRINT"AND <M>ANUAL AND YOU WILL BE
ASKED TO SELECT ONE.":PRINT
1720 PRINT"AUTOMATIC OPERATION"
1730 PRINT"    THE COMPUTER WILL ASK YOU
 HOW MANY DISCS YOU WOULD LIKE"
1740 PRINT"MOVED AND HOW FAST YOU WOULD
LIKE THEM MOVED.  IT WILL"
1750 PRINT"THEN PROCEED TO TRANSFER THE
DISCS FROM PEG 1 TO PEG 3"
1760 PRINT"USING PEG 2 AS AN INTERMEDIAT
E."
1770 GOSUB 2020
1780 LOCATE30,3:PRINT"PLAYING HANOI"
1790 LOCATE30,4:PRINT"-------------":PRI
NT
1800 PRINT"MANUAL OPERATION"
1810 PRINT"    HERE YOU WILL BE ASKED HO
W MANY DISCS YOU WOULD LIKE"
1820 PRINT"TO MOVE.  THEN YOU WILL BE PR
OMPTED TO TYPE THE NUMBER OF"
1830 PRINT"THE PEG THE DISC IS TO BE TAK
EN OFF AND THE NUMBER OF THE"
1840 PRINT"PEG THE DISC IS TO BE PLACED
ON.  WHEN YOU HAVE SUCCESSFULLY"
1850 PRINT"TRANSFERRED ALL DISCS YOU WIL
L BE TOLD HOW MANY MOVES"
1860 PRINT"IT TOOK."
1870 GOSUB2020
1880 LOCATE30,3:PRINT"SUGGESTION"
1890 LOCATE30,4:PRINT"----------":PRINT
1900 PRINT"    IF YOU ARE UNSURE OF THE
GAME TRY AUTOMATIC OPERATION"
1910 PRINT"WITH ABOUT FOUR DISCS AND LOW
 SPEED (E.G. 10).  THEN SIT"
1920 PRINT"BACK AND LET THE COMPUTER DO
ALL THE WORK - AFTER ALL THAT"
1930 PRINT"IS WHAT IT IS THERE FOR.":PRI
NT
1940 PRINT"    FOR A GIVEN NUMBER OF DIS
CS, N, THE SMALLEST NUMBER OF"
1950 PRINT"MOVES REQUIRED IS GIVEN BY :-
"
1960 PRINT"              M = 2[N-1
1970 PRINT"SO FOR 3 DISCS THAT'S 7 MOVES
, 4 DISCS 15 MOVES UP TO"
1980 PRINT"THE MAXIMUM OF 10 DISCS WHICH
 REQUIRES 1023 MOVES.":PRINT
1990 PRINT"    HAVE FUN!"
2000 GOSUB 2020
2010 RETURN
2020 LOCATE7,18:PRINT"PRESS <RETURN> WHE
N READY";
2030 INPUT D$:CLS
2040 RETURN
2050 IN$=INKEY$:IFIN$=""THEN2050ELSEIN=V
AL(IN$):RETURN
```

**** REGISTER DISPLAY ****

HITACHI PEACH

```
10 '******* REGISTER DISPLAY PROGRAM ***
****
20 '
30 '
40 'Program loads display routine in mac
hine
50 'language format at selected  locatio
n in
60 'User RAM or a default location at
6F90
70 '(top of User RAM).
80 '
90 'The program runs after loading and g
ives
100 'a sample display.
110 '
120 SCREEN 0,,0:CLEAR 300,&H6F00
130 PRINT"Insert RAM store location :xxx
x :or DEFT"
140 A$=INPUT$(4)
150 IF A$="DEFT" THEN A$="&H6F90" ELSE A
$="&H"+A$
160 POKE &H0107,VAL(LEFT$(A$,4))
170 POKE &H0108,VAL("&H"+(RIGHT$(A$,2)))
+15
180 FOR I=0 TO 103
190 READ B$
```

```
200 POKE VAL(A$)+I,VAL("&H"+B$)
210 NEXT I
220 DATA 86,7E,B7,01,06,3F,39,3F,BD,A4,4
0,3F,BD
230 DATA DC,E6,33,6C,34,40,1F,43,86,53,B
D,E8,20
240 DATA 8D,40,A6,C0,BD,DD,32,A6,C0,BD,D
D,32,BD
250 DATA B0,BE,8E,DE,20,BD,DE,31,A6,C0,B
D,DD,32
260 DATA 8D,1D,A6,84,81,58,26,F0,81,45,2
7,23,BD
270 DATA DE,31,A6,C0,BD,DD,32,A6,C0,BD,D
D,32,8D
280 DATA 0A,A6,84,20,E9,BD,B1,16,BD,B1,1
6,BD,B1
290 DATA 16,BD,B1,16,BD,B1,16,39,35,40,3
5,FF,00
300 EXEC VAL(A$)
310 END
```

```
         ORG     $6790
         LDA     #$7E     INITIALISE SWI VECTOR IN C/W
         STA     >$0106   BASIC LINES 160 AND 170
         SWI              RETURN TO PROGRAM
         RTS
         SWI
         JSR     >$A440   RETURN TO BASIC
         SWI
         JSR     >$DCE6   RETURN TO MON
         LEAU    12,S     CALCULATE ORIGINAL STACK POINTER
         PSHS    U        SAVE IT
         TFR     S,U      STACK POINTER: S TO U
         LDA     #$53     CHARACTER S
         JSR     >$E820   PRINT S
         BSR     SPC2     2 SPACES
         LDA     ,U+
         JSR     >$DD32   2 BYTES FROM STACK
         LDA     ,U+      AND PRINT
         JSR     >$DD32
         JSR     >$B0BE   CR/LF
         LDX     #$DE20   POINT TO REGISTER HEADINGS TABLE
RDP1     JSR     >$DE31   PRINT HEADINGS AND SPACE
         LDA     ,U+
         JSR     >$DD32   1 BYTE FROM STACK AND PRINT
         BSR     SPC5     TO 6FE3: 5 SPACES
         LDA     ,X
         CMPA    #$58     TEST FOR END OF 1 BYTE REGS.
         BNE     RDP1     TO 6FBC
RDP2     CMPA    #$45     TESTS FOR END OF HEADINGS
         BEQ     EXIT     TO 6FF3 AND EXIT
         JSR     >$DE31   PRINTS REG. HEADING AND SPACE
         LDA     ,U+
         JSR     >$DD32   PRINTS 2 BYTES FROM STACK
         LDA     ,U+
         JSR     >$DD32
         BSR     SPC2     TO 6FEC: 2 SPACES
         LDA     ,X
         BRA     RDP2     TO 6FCC
SPC5     JSR     >$B116
         JSR     >$B116
         JSR     >$B116
SPC2     JSR     >$B116
         JSR     >$B116
         RTS
EXIT     PULS    U        DISCARD SP
         PULS    CC,A,B,DP,X,Y,U,PC      RESTORE REGS. AND SP
         END
```

****** L2/16K DESERT CHASE ******

TRS-80/SYSTEM-80

```
10 REM ------ WRITTEN BY: ANDREW N. DOWNEY ------
20 CLS:RANDOM:PRINT@0,CHR$(23);"PLEASE STAND BY    :"
30 CLEAR(1300):DIM C$(51):DEFINTA-Z:GOSUB1000:R=RND(100)+200
40 CLS:PRINTTAB(25);"DESERT CHASE":PRINTTAB(25);"====== =====";:
PRINT@256,"WOULD YOU LIKE INSTRUCTIONS";
50 K$=INKEY$:IFK$=""THEN50ELSEIFK$="N"THEN120
60 PRINT@128,CHR$(30);
70 PRINT"
WELCOME TO 'DESERT CHASE'.  THE OBJECT IS TO TRAVEL ACROSS THE
VAST SIMPSON DESERT.   THE DISTANCE VARIES BETWEEN 200 MILES &
300 MILES.   A LARGE TRIBE OF HOSTILE NATIVES IS RUMORED TO BE
IN EXISTENCE AND IS SURE TO CHASE YOU."
80 PRINT"
YOU HAVE BEEN GIVEN A SINGLE HUMP CAMEL - CHARLES C. CAMEL -
TO HELP YOU IN YOUR VENTURE."
90 PRINT"
WHEN ASKED FOR COMMANDS THE CURSOR WILL FLASH,  AT THIS POINT,
PRESS KEY CORRESPONDING TO THE NUMBER REQUIRED."
100 PRINT:PRINT"PRESS ANY KEY TO CONTINUE"
110 K$=INKEY$:IFK$=""THEN110
120 CLS:PRINT@0,"D E S E R T - C H A S E
   C O M M A N D S :
#1 DRINK FROM WATERBAG
#2 CONTINUE SLOWLY
#3 AHEAD FULL SPEED
#4 REST FOR THE NIGHT
#5 CHECK SUPPLIES
#6 WAIT FOR HELP"
130 PRINT@512,"
YOU HAVE RENEWED YOUR WATER SUPPLY AT NEARBY WATER-HOLE AND HAVE
ONE LITRE OF WATER.  THIS WILL LAST YOU SIX DRINKS.  IF FOUND BY
HELP THEY WILL GIVE YOU  1/2  LITRE.   IF HELP DOES NOT FIND YOU
AFTER COMMAND 6, --- YOU DIE --- !!!"
```

```
140 PRINT"THE MINISTRY OF THE INTERIOR AND I WISH YOU GOOD LUCK
!!!"
150 GOSUB940
160 IFC>RTHEN630
170 Z=Z-1:IFZ=1PRINT"--------- WARNING --------- YOU NEED A DRIN
K."
180 IFZ<0 THEN 880 ELSE P=P+1
190 X2=RND(10)+2.5:IFQ>0THEN510
200 IFP<4THEN250
210 C1=C1+X2:IFC1<CTHEN240
220 FORD=1TO1000:NEXT:PRINT@512,CHR$(31);"
THE NATIVES HAVE CAPTURED YOU.  CAMEL AND PEOPLE SOUP IS"
230 PRINT"THEIR FAVORITE DISH!!!!":C4=7:GOSUB980:GOTO 840
240 PRINT"THE NATIVES ARE";C-C1;"MILES BEHIND YOU"
250 IF C=0 THEN 260 ELSE PRINT@101,C;"DOWN -";R-C;"TO GO.";
260 IFT2<1THENC4=10
270 GOSUB980:GOSUB950:T2=1:PRINT@38,"ENTER YOUR COMMAND";
280 Y$=INKEY$:IFY$<>""THEN290ELSEPRINT@58,CHR$(140);:FORD=1TO100
:NEXT:PRINT@58,CHR$(128);:FORD=1TO100:NEXT:GOTO280
290 IF ASC(Y$)<49 OR ASC(Y$)>54 THEN 280 ELSE Y=VAL(Y$)
300 FORD=1TO200:NEXT:PRINT@56,"  ";:PRINT@512,CHR$(31);
310 ON Y GOTO 460,340,380,410,430
320 T=RND(10):IFT<>1THEN740
330 C4=37:GOSUB980:PRINT@512,CHR$(31);"
HELP HAS FOUND YOU IN A STATE OF UNCONSCIOUSNESS.":S=3:Z=4:GOTO1
60
340 F=F+1:IFF=8THEN620
350 GOSUB 480
360 X1=RND(10):C=C+X1:C4=1:GOSUB980
370 PRINT@512,CHR$(31);"
YOUR CAMEL LIKES THIS PACE":GOTO 160
380 F=F+3:IFF>7THEN620
390 GOSUB 480:X1=2*RND(10):C=C+X1:C4=22:GOSUB980
400 PRINT@512,CHR$(31);"
YOUR CAMEL IS BURNING ACROSS THE DESERT SANDS":PRINT:GOTO 160
410 C4=10:GOSUB980:PRINT@512,CHR$(31);"
YOUR CAMEL THANKS YOU!"
420 F=0:GOTO170
430 PRINT"
yOUR CAMEL HAS ";7-F;" GOOD DAYS LEFT."
440 PRINT"YOU HAVE ";S;" DRINKS LEFT IN YOUR WATER-BAG."
450 PRINT"YOU CAN GO ";Z;" COMMANDS WITHOUT DRINKING.":GOTO170
460 S=S-1:C4=31:GOSUB980:IFS<0PRINT@512,;:GOTO740
470 PRINT@512,"
BETTER WATCH FOR A WATER-HOLE !":Z=4:GOTO260
480 A=RND(100):IFA>5THEN600
490 C4=4:GOSUB980
500 PRINT@512,"WILD TRIBESMEN HIDDEN IN THE SAND HAVE CAPTURED Y
OU.
LUCKILY THE LOCAL ELDER HAS AGREED TO THEIR RANSOM
DEMANDS ....... BUT ....... WATCH FOR THE NATIVES!!!
YOU HAVE A NEW CHOICE OF SUB-COMMANDS  :
"
510 PRINT"#7 ATTEMPT ESCAPE --- ";
520 INPUT"#8 WAIT FOR PAYMENT --- SUB COMMAND ";X:IFX<7ORX>8THEN
500
530 PRINT@512,CHR$(31);:IFX=8THEN570
540 X1=RND(10):IFX1<5THEN560
550 PRINT"
CONGRATULATIONS, YOU  SUCCESSFULLY  ESCAPED!!!":Q=0:GOTO160
560 PRINT"YOU WERE MORTALLY WOUNDED BY A SPEAR WHILE ESCAPING.":
C4=34:GOSUB980:GOTO840
570 X1=RND(100):IFX1>24THEN590
580 PRINT"
YOUR RANSOM HAS BEEN PAID AND YOU ARE FREE TO GO.":Q=0:GOTO160
590 PRINT"
THE LOCAL ELDERS ARE COLLECTING ... JUST WAIT ...":FORD=1TO1000:
NEXT
600 A=RND(10):IFA>2THEN650
610 PRINT@512,CHR$(31);"
YOU HAVE ARRIVED AT A WATER-HOLE....YOUR CAMEL
EATS HAPPILY WHILE YOU REFILL YOUR WATER-BAG":C4=13:GOSUB980:FOR
D=1TO2500:NEXT:Z=4:S=6:RETURN
620 C4=16:GOSUB980:PRINT@512,"
YOU IDIOTIC FOOL !!! YOU RAN YOUR POOR CAMEL TO DEATH!!":GOTO740
630 PRINT@38,"YOU TRAVELLED ";C;"MILES.";:PRINT@102,"VERY  WELL
 DONE !!!";
640 PRINT@512,CHR$(31);"
YOU WIN, A  PARTY IS BEING GIVEN IN YOUR HONOR.......
........ THE NATIVES ARE PLANNING TO  ATTEND ........":C4=28:GOS
UB980:GOTO840
650 X1=RND(100):IFX1>5THEN720
660 C4=19:GOSUB980:PRINT@167,C$(C4);:PRINT@423,C$(C4);
670 PRINT@512,"
YOU HAVE BEEN CAUGHT IN A SANDSTORM....GOOD LUCK!":FORD=1TO1000:
NEXT
680 X5=RND(100):X6=RND(100):IFX6<50THEN710
690 C=C+X5:GOTO710
700 C=C-X5
710 PRINT"YOUR NEW POSITION IS ";C;" MILES SO FAR!":FORD=1TO1500
:NEXT:PRINT@167,STRING$(23,131);:PRINT@423,STRING$(23,176);:RETU
RN
720 X1=RND(100):IFX1>5RETURN
730 C1=C1+1:C4=25:GOSUB980:PRINT@512,CHR$(31);"
YOUR CAMEL HURT HIS HUMP.
LUCKILY THE NATIVES WERE FOOTWEARY!!!":FORD=1TO1500:NEXT:RETURN
740 U=RND(10):PRINT"
YOU DIED IN THE DESERT.":IFF>7GOTO840
750 IFU>1THEN770
760 PRINT"THE NATIONAL CAMEL'S UNION IS NOT ATTENDING YOUR FUNER
AL":C4=49:GOSUB980:GOTO840
770 IF U>3 THEN 790
780 PRINT"YOUR BODY WAS EATEN BY VULTURES AND IMPORTED CANNIBALS
":C4=4:GOSUB980:GOTO840
790 IF U>5 THEN 810
800 PRINT"THE LOCAL ELDER NOW USES YOUR SKULL FOR A MONEY PURSE!
!!":C4=43:GOSUB980:GOTO840
810 IF U>7 THEN 830
820 PRINT"PEOPLE WITH LITTLE INTELLIGENCE SHOULD STAY OUT OF THE
 DESERT":C4=46:GOSUB980:GOTO840
```

```
830 PRINT"TURKEYS SHOULD FLY, NOT RIDE CAMELS":C4=40:GOSUB980
840 PRINT@960,"WANT A NEW CAMEL AND A NEW GAME ";
850 K$=INKEY$:IFK$=""THEN850
860 IFK$="Y"THENT2=0:GOTO120ELSE870
870 IFK$<>"N"THEN850ELSE890
880 PRINT@512,"YOU DIED OF DEHYDRATION ...... DIDN'T DRINK ENOUG
H.":GOTO740
890 FOR C2=1TO10
900 CLS:PRINTCHR$(23):PRINT"................................"
910 PRINT"              CHICKEN!!"
920 PRINT"................................"
930 FORD=1TO500:NEXT:NEXT:END
940 Z=4:S=6:C=0:C1=0:Q=0:F=0:P=0:RETURN
950 FORC2=15527TO15549:POKEC2,131:POKEC2+256,176:NEXTC2
960 FORC2=15526TO15782STEP64:POKEC2,191:POKEC2+24,191:NEXTC2
970 RETURN
980 PRINT@231,C$(C4);:PRINT@295,C$(C4+1);:PRINT@359,C$(C4+2);
990 RETURN
1000 L1$=STRING$(23,128):L2$=STRING$(2,128):L3$=STRING$(3,128):L
4$=STRING$(4,128):L5$=STRING$(5,128):L6$=STRING$(6,128):L7$=STRI
NG$(7,128):L8$=STRING$(8,128):L9$=STRING$(9,128)
1010 C$(1)=L9$+CHR$(160)+CHR$(176)+CHR$(176)+CHR$(128)+CHR$(128)
+CHR$(191)+CHR$(143)+CHR$(141)+L6$
1020 C$(2)=L7$+CHR$(160)+CHR$(184)+STRING$(3,191)+CHR$(189)+CHR$
(176)+CHR$(159)+L8$
1030 C$(3)=L7$+CHR$(129)+CHR$(170)+CHR$(181)+CHR$(128)+CHR$(128)
+CHR$(170)+CHR$(181)+L9$
1040 C$(4)=L1$:C$(5)=L1$:C$(6)=L1$
1050 C$(7)=L8$+CHR$(182)+CHR$(140)+CHR$(185)+CHR$(160)+CHR$(156)
+CHR$(188)+CHR$(172)+CHR$(144)+L7$
1060 C$(8)=L5$+CHR$(172)+CHR$(188)+CHR$(176)+CHR$(176)+CHR$(191)
+CHR$(176)+CHR$(176)+CHR$(179)+CHR$(191)+CHR$(179)+CHR$(176)+CHR
$(188)+CHR$(156)+L5$
1070 C$(9)=L6$+CHR$(139)+CHR$(143)+CHR$(191)+STRING$(5,143)+CHR$
(191)+CHR$(143)+CHR$(135)+L6$
1080 C$(10)=L1$:C$(11)=C$(1):C$(12)=C$(2)
1090 C$(13)=L9$+CHR$(160)+CHR$(176)+CHR$(144)+L5$+CHR$(176)+CHR$
(188)+CHR$(188)+CHR$(179)+CHR$(155)+CHR$(191)
1100 C$(14)=CHR$(160)+CHR$(140)+CHR$(176)+CHR$(140)+CHR$(144)+L2
$+CHR$(160)+CHR$(184)+STRING$(3,191)+CHR$(189)+CHR$(176)+CHR$(17
6)+CHR$(188)+CHR$(143)+CHR$(131)+L2$+CHR$(130)+CHR$(128)+CHR$(19
1)
1110 C$(15)=CHR$(129)+CHR$(128)+CHR$(191)+CHR$(128)+CHR$(130)+L2
$+CHR$(129)+CHR$(170)+CHR$(181)+L2$+CHR$(170)+CHR$(181)+L2$+CHR$
(176)+STRING$(4,188)+CHR$(176)+CHR$(191)
1120 C$(16)=L6$+"R . I . P ."+L4$
1130 C$(17)=L8$+CHR$(160)+CHR$(176)+CHR$(176)+L7$+CHR$(176)+L2$
1140 C$(18)=L4$+CHR$(160)+CHR$(176)+CHR$(176)+CHR$(184)+STRING$(
3,191)+CHR$(189)+CHR$(176)+CHR$(176)+CHR$(188)+CHR$(188)+CHR$(17
6)+CHR$(131)+CHR$(191)+CHR$(131)+CHR$(128)
1150 C$(19)=STRING$(23,191):C$(20)=C$(19):C$(21)=C$(19)
1160 C$(22)=L7$+CHR$(160)+CHR$(176)+CHR$(176)+L5$+CHR$(176)+CHR$
(188)+CHR$(188)+CHR$(176)+L4$
1170 C$(23)=L5$+CHR$(164)+CHR$(184)+STRING$(3,191)+CHR$(189)+CHR
$(176)+CHR$(176)+CHR$(188)+CHR$(143)+CHR$(131)+L7$
1180 C$(24)=L6$+CHR$(130)+CHR$(173)+CHR$(144)+CHR$(160)+CHR$(158
)+CHR$(129)+L4$+L7$
1190 C$(25)=CHR$(128)+CHR$(191)+CHR$(143)+CHR$(131)+CHR$(131)+CH
R$(143)+CHR$(191)+L4$+CHR$(160)+STRING$(3,176)+CHR$(184)+CHR$(18
8)+CHR$(148)+L3$
1200 C$(26)=CHR$(128)+CHR$(191)+CHR$(188)+CHR$(176)+CHR$(176)+CH
R$(188)+CHR$(191)+L3$+CHR$(176)+CHR$(189)+CHR$(188)+CHR$(188)+CH
R$(190)+CHR$(176)+CHR$(187)+CHR$(191)+CHR$(148)+L2$
1210 C$(27)=L9$+CHR$(130)+CHR$(170)+CHR$(181)+L2$+CHR$(170)+CHR$
(181)+L5$
1220 C$(28)=L4$+"CONGRATULATIONS!"+L3$:C$(29)=C$(28):C$(30)=C$(2
8)
1230 C$(31)=L6$+STRING$(8,191)+CHR$(189)+CHR$(176)+CHR$(144)+L6$
1240 C$(32)=L6$+STRING$(8,191)+CHR$(159)+CHR$(131)+CHR$(145)+L6$
1250 C$(33)=L8$+L8$+CHR$(132)+L6$
1260 C$(34)=L5$+L5$+CHR$(170)+L6$+L6$
1270 C$(35)=L5$+L5$+CHR$(186)+CHR$(144)+L6$+CHR$(176)+CHR$(188)+
CHR$(176)+L2$
1280 C$(36)=L6$+CHR$(180)+CHR$(176)+CHR$(176)+CHR$(188)+CHR$(190
)+CHR$(188)+CHR$(188)+CHR$(176)+CHR$(188)+CHR$(188)+STRING$(3,12
8)+CHR$(191)+L3$
1290 C$(37)=L1$
1300 C$(38)=L4$+CHR$(160)+CHR$(176)+L3$+STRING$(5,176)+CHR$(144)
+CHR$(128)+CHR$(144)+CHR$(176)+CHR$(176)+L3$
1310 C$(39)=L4$+CHR$(170)+CHR$(191)+STRING$(3,188)+STRING$(5,191
)+CHR$(189)+CHR$(188)+CHR$(191)+CHR$(189)+CHR$(191)+L3$
1320 C$(40)=L1$
1330 C$(41)=L7$+CHR$(176)+CHR$(188)+STRING$(3,191)+CHR$(188)+CHR
$(176)+CHR$(128)+CHR$(188)+CHR$(191)+CHR$(140)+L5$
1340 C$(42)=L5$+STRING$(5,131)+CHR$(191)+CHR$(179)+CHR$(179)+STR
ING$(4,131)+L6$
1350 C$(43)=L9$+CHR$(160)+CHR$(128)+CHR$(179)+CHR$(128)+CHR$(144
)+L9$
1360 C$(44)=L8$+CHR$(184)+CHR$(143)+CHR$(189)+CHR$(176)+CHR$(190
)+CHR$(143)+CHR$(180)+L8$
1370 C$(45)=L8$+CHR$(139)+CHR$(191)+CHR$(183)+CHR$(179)+CHR$(187
)+CHR$(191)+CHR$(135)+L8$
1380 C$(46)=L4$+STRING$(15,188)+L4$
1390 R2=RND(9)+9:C$(47)=L4$+CHR$(191)+CHR$(191)+" I.Q. ="+STR$(R
2)+CHR$(128)+CHR$(191)+CHR$(191)+L4$
1400 C$(48)=L4$+STRING$(15,143)+L4$
1410 C$(49)=L6$+STRING$(9,176)+CHR$(144)+L5$
1420 C$(50)=L6$+CHR$(191)+" N.C.U."+CHR$(170)+CHR$(149)+L5$
1430 C$(51)=L6$+STRING$(4,131)+CHR$(171)+CHR$(151)+STRING$(3,131
)+CHR$(129)+L5$
1440 RETURN
```

****** L2/16K   FORMATION ******

TRS-80/SYSTEM-80

```
00100 DELAY     EQU     0060H     ;DELAY LOOP
00200 AMTDLY    EQU     1300H     ;AMOUNT OF DELAY
```

```
WRITE   EQU     0264H   ;WRITES BYTE TO TAPE
DISPLY  EQU     28A7H   ;DISPLAYS MESSAGE ON SCREEN
SCREEN  EQU     3C00H   ;START OF SCREEN MEM
ENDSCR  EQU     3FFFH   ;END OF SCREEN MEM
ENDPRG  EQU     40F9H   ;END OF BASIC PROGRAM POINTER
CLS     EQU     01C9H   ;CLEARS SCREEN
GETCHR  EQU     002BH   ;GETS KEY DEPRESSED
GETVAL  EQU     1BB3H   ;GETS INPUT VALUE
RECON   EQU     0212H   ;TURNS RECORDER ON
RECOF   EQU     01F8H   ;  "      "     OFF
LEDER   EQU     0287H   ;WRITES LEADER&SYNCH TO TAPE
OMERR   EQU     19A7H   ;OUT OF MEMORY ERROR
OPENCM  EQU     417AH   ;'OPEN' COMMAND POINTER
        ORG     433FH   ;LOW MEMORY START
FIRST   NOP
MEMSTR  DEFS    1024D   ;SPACE TO HOLD FORM
MESS1   DEFM    '       * F O R M A T I O N *'
        DEFB    0AH
        DEFM    'COPYRIGHT 1982 DAVID R. GRIGG'
        DEFW    0A0AH
        DEFM    'OPTIONS'
        DEFB    0AH
        DEFB    0AH
MESS0   DEFM    '1. MODIFY EXISTING FORM
        DEFB    0AH
MESS2   DEFM    '2. CREATE NEW FORM'
        DEFB    0AH
MESS3   DEFM    '3. STORE NEWLY-CREATED FORM'
        DEFB    0AH
MESS4   DEFM    '4. FILL EXISTING FORMS'
        DEFB    0AH
MESS5   DEFM    '5. RETURN TO BASIC'
        DEFB    0AH
        DEFB    00H
BEGIN   LD      HL,OPTION       ;INITIALISATION ROUTINE
        LD      (OPENCM),HL     ;SETS UP 'OPEN' COMMAND
        LD      HL,LAST         ;AS INTRO TO FORMAT
        XOR     A               ;FROM BASIC
        LD      (HL),A          ;ZEROS LAST BYTE OF PROG
        INC     HL              ;AND SPACES BEYOND FOR
        LD      (HL),A          ; NEW BASIC START
        LD      (40A4H),HL      ;RESET START OF BASIC
        INC     HL
        LD      (HL),A
        INC     HL
        LD      (40F9H),HL      ;RESET END OF BASIC
        LD      (40FBH),HL      ;RESET ARRAY POINTER
        LD      (40FDH),HL      ;RESET FREE SPACE PTR
        LD      IY,YTAB         ;POINT IY TO TABLE
        LD      A,7DH
        LD      (IY+3),A        ;INITIALISE LINE NO
        XOR     A               ; FOR DATA LINES TO
        LD      (IY+2),A        ; 32000
        JP      06CCH   ;BACK TO BASIC
;
NAME    DEFM    'FORMAT'        ;NAME FOR TAPE COPY
MESS6   DEFM    'IS RECORDER READY'
        DEFB    00H
OPTION  LD      IX,XTAB         ;SET IX TO TABLE
        LD      IY,YTAB         ;SET IY TO TABLE
        CALL    CLS             ;CLEAR SCREEN
        LD      HL,MESS1        ;DISPLAY OPTION LIST
        CALL    DISPLY
TST     CALL    GETVAL          ;GET NO OF OPTION CHOSEN
        RST     10H
        CALL    VALACC          ;EVAULATES NO.
        CP      01H
        JP      Z,MODFRM        ;MODIFY FORM
        CP      02H
        JP      Z,MKFORM        ;CREATE FORM
        CP      03H
        JP      Z,TAPE          ;STORE ON TAPE
        CP      04H
        JP      Z,FILL          ;FILL FORMS
        CP      05H
        JP      Z,06CCH ;BACK TO BASIC
        JP      OPTION
; END OF CALLING PROGRAM
SC2MEM  LD      BC,1024D        ;ROUTINE STORES SCREEN
        LD      HL,SCREEN       ; DISPLAY IN MEMORY
        LD      DE,MEMSTR
        LDIR
        RET
;
MEM2SC  LD      BC,1024D        ;ROUTINE RETURNS STORED
        LD      HL,MEMSTR       ; FORM FROM MEMORY TO
        LD      DE,SCREEN       ; SCREEN.
        LDIR
        RET
;
MODFRM  CALL    MEM2SC          ;GET EXISTING FORM
        JP      MKFM02
MKFORM  CALL    CLS             ;CLEAR SCREEN
MKFM02  LD      HL,SCREEN       ;SET HL TO START OF SCREEN
LOOP01  CALL    GETKEY          ;GET KEY BEING DEPRESSED
TST2    CP      32D             ;IS IT A CONTROL CHAR?
        JP      M,CONTRL        ; IF LESS THAN 32, CONTROL
        CP      91D             ;IS IT AN UP ARROW?
        JP      Z,UPARR
        CP      40H     ;@ KEY FOR CURSOR SYMBOL
        JP      Z,ASYMBL
SKIP01  LD      (HL),A  ;DISPLAY CHARACTER ON SCREEN
        INC     HL
        CALL    TESTHI  ;ARE WE OVER END OF SCREEN?
        JP      LOOP01
;
UPARR   CALL    REPLCE  ;RELACE CHAR COVERED BY CURSOR
        LD      DE,0040H
        XOR     A
        SBC     HL,DE   ;MOVE SCREEN POINTER UP ONE LINE
```

```
                CALL    TESTLO  ;ARE WE BEYOND START OF SCREEN?
                JP      LOOP01
      ;
      CONTRL    CP      08H     ;IS IT BACK ARROW?
                JP      Z,BACKSP
                CP      0AH     ;IS IT DOWN ARROW?
                JP      Z,DOWNAR
                CP      09H     ;IS IT FORWARD ARROW?
                JP      Z,FWDARR
                CP      0DH     ;IS IT CARRIAGE RET.?
                JP      Z,CARRET
                CP      1BH     ;SHIFT UP ARROW?
                JP      Z,CLEAR
                JP      LOOP01
      ;
      ;
      ASYMBL    LD      A,5FH   ;TURN @ INTO CURSOR SYMBOL
                JP      SKIP01
      ;

      REPLCE    LD      A,(IY)  ;GET STORED CHARACTER
                LD      (HL),A  ;PUT IT BACK ON SCREEN
                RET
      ;
      BACKSP    CALL    REPLCE
                DEC     HL      ;MOVE POINTER BACK
                CALL    TESTLO  ;OVER START OF SCREEN?
                JP      LOOP01
      ;
      DOWNAR    CALL    REPLCE
                LD      DE,64D  ;INCREMENT SCREEN POINTER ONE LINE
                ADD     HL,DE
                CALL    TESTHI  ;OVER END OF SCREEN?
                JP      LOOP01
      ;
      FWDARR    CALL    REPLCE
                INC     HL      ;INCREMENT SCREEN POINTER
                CALL    TESTHI  ;OVER END OF SCREEN?
                JP      LOOP01
      ;
      TESTHI    LD      A,H
                CP      40H     ;GREATER THAN 3FFFH?
                RET     M
                LD      HL,ENDSCR       ;IF SO, SET TO 3FFFH
                RET
      ;
      TESTLO    LD      A,H
                CP      3CH     ;LESS THAN 3C00H?
                RET     P
                LD      HL,SCREEN       ;IF SO, SET TO 3C00H
                RET
      ;
      CARRET    CALL    REPLCE
                LD      DE,64D  ;MOVE POINTER UP ONE LINE
                ADD     HL,DE
                LD      A,L
                AND     192D    ;AND TO START OF THAT LINE
                LD      L,A
                CALL    TESTHI  ;NOT OVER END OF SCREEN?
                JP      LOOP01
      ;
      TAPE      CALL    CLS
                LD      HL,MESS6        ;CHECK RECORDER O.K.
                CALL    DISPLY
                CALL    GETVAL
                XOR     A
                CALL    RECON   ;TURN ON RECORDER
                LD      HL,LAST
                LD      DE,FIRST ;SET TO BOUNDARIES OF PROGRAM
                XOR     A
                SBC     HL,DE   ;NO OF BYTES TO MOVE
                INC     HL
                CALL    LEDER   ;WRITE LEADER & SYNCH BYTE
                LD      A,55H
                CALL    WRITE
                LD      B,06H
                PUSH    HL
                LD      HL,NAME ;PROGRAM NAME IS 'FORMAT'
      LOOP40    LD      A,(HL)
                CALL    WRITE   ;WRITE NAME TO TAPE
                INC     HL
                DJNZ    LOOP40
                POP     HL
      LOOP41    DEC     H       ;DECREASE HL BY 256BYTES
                JP      M,SKIP40  ;LAST BLOCK IF MINUS.
                LD      A,3CH
                CALL    WRITE
                XOR     A
                CALL    WRITE   ;256 BYTES IN BLOCK
                CALL    WRTBTE  ;WRITE THESE BYTES
                JR      LOOP41
      SKIP40    XOR     A       ;INCOMPLETE BLOCK
                CP      L
                JR      Z,SKIP41  ;IF L ZERO, NO BYTES LEFT
                LD      A,3CH
                CALL    WRITE
                LD      A,L     ;NO OF BYTES IN THIS BLOCK
                CALL    WRITE
                CALL    WRTBTE  ;WRITE THESE BYTES
      SKIP41    LD      A,78H   ;END OF TAPE DATA
                CALL    WRITE
                LD      BC,BEGIN
                LD      A,C
                CALL    WRITE   ;WRITE OPERATIONAL START
                LD      A,B     ; OF PROGRAM
                CALL    WRITE
                CALL    RECOF   ;TURN OFF RECORDER
                JP      OPTION  ;GO BACK FOR NEXT OPTION
      WRTBTE    LD      B,A     ;SAVE A
                LD      A,E
```

```
22300           CALL    WRITE   ;WRITE START ADDRESS OF
22400           LD      A,D     ; THIS BLOCK
22500           CALL    WRITE
22600           ADD     A,E     ;ADD FOR CHECKSUM
22700           LD      C,A
22800 LOOP42    LD      A,(DE)
22900           CALL    WRITE   ;WRITE THIS BYTE TO TAPE
23000           ADD     A,C     ;ADD FOR CHECKSUM
23100           LD      C,A
23200           INC     DE
23300           DJNZ    LOOP42  ;KEEP GOING
23400           LD      A,C
23500           CALL    WRITE   ;WRITE CHECKSUM TO TAPE
23600           RET
23700 ;
23800 ;
23900 FILL      CALL    MEM2SC; GET CLEAN FORM
24000           LD      B,01H   ;SET B TO FIRST ENTRY FIELD
24100           CALL    GETFLD  ;GET SCREEN ADDRESS OF FIELD
24200 LOOP07    LD      A,(HL)
24300           LD      (IY),A  ;SAVE CURRENT CHARACTER
24400 LOOP08    CALL    GETKEY  ;GET KEY BEING PRESSED
24500           CP      32D     ;IS IT A CONTROL CHAR?
24600           JP      M,CNT01
24700           CP      91D     ;AN UP ARROW?
24800           JP      Z,UPAR01
24805           CP      96D     ;SHIFT @?
24810           JP      Z,CLER01
24900           LD      (HL),A  ;OTHERWISE, PUT CHAR ON SCREEN
25000           INC     C       ;C IS NO OF CHARS PUT IN THIS
25100           INC     HL      ; ENTRY FIELD SO FAR
25200           LD      A,(IY+1);GET LENGTH OF CURRENT FIELD
25300           CP      C       ;HAVE WE EXCEEDED IT?
25400           JP      Z,DNAR02;IF SO, JUMP TO NEXT FIELD
25500           JP      LOOP08
25600 ;
25700 UPAR01    CALL    REPLCE
25800 UPAR02    DEC     B       ;GO BACK TO LAST FIELD
25900           JP      NZ,LOOP10
26000           LD      B,(IX)  ;IF BEYOND FIRST FLD, GET LAST.
26100 LOOP10    CALL    GETFLD
26200           JP      LOOP07
26300 ;
26400 DNAR01    CALL    REPLCE
26500 DNAR02    INC     B       ;GO TO NEXT FIELD
26600           LD      A,(IX)
26700           CP      B       ;ARE WE OVER TOTAL FIELDS?
26800           JP      P,LOOP11
26900           LD      B,01H   ;IF SO, SET TO FIRST FIELD.
27000 LOOP11    CALL    GETFLD
27100           JP      LOOP07
27200 ;
27300 BKAR01    CALL    REPLCE
27400           DEC     HL      ;DECREMENT SCREEN POINTER
27500           DEC     C       ;DEC. CHARS SO FAR IN FIELD
27600           JP      NZ,LOOP08
27700           JP      UPAR02  ;IF BEYOND START OF FLD, GO TO
27800 ;                                  LAST FIELD
27900 FWAR01    CALL    REPLCE
28000           INC     HL      ;INC. SCREEN POINTER.
28100           INC     C       ;INC CHARS IN FIELD
28200           LD      A,(IY+1); MAX CHARS IN FIELD
28300           CP      C       ; IF OVER, GO TO NEXT
28400           JP      M,DNAR01        ;FIELD
28500           JP      LOOP08
28600 ;
28700 GETFLD    LD      C,01H   ;C HOLDS CHARS SO FAR IN FLD
28800           PUSH    IX
28900           POP     DE      ;DE NOW HAS START OF TABLE
29000           INC     DE
29100           PUSH    BC      ;SAVE BC
29200 LOOP12    DEC     B       ;B IS INDEX TO FIELD WANTED
29300           JP      Z,THERE ;IF IT'S ZERO, WE'VE REACHED IT.
29400           INC     DE
29500           INC     DE
29600           INC     DE      ;JUMP UP TO NEXT TABLE ENTRY
29700           JR      LOOP12
29800 THERE     LD      A,(DE)
29900           LD      L,A     ;SET HL TO ADDRESS FROM TABLE
30000           INC     DE
30100           LD      A,(DE)
30200           LD      H,A     ;HL NOW HAS SCREEN ADDRESS
30300           INC     DE
30400           LD      A,(DE)  ;LENGTH OF FIELD
30500           LD      (IY+1),A        ;STORE IN IY+1
30600           POP     BC ;RETURN BC
30700           RET
30800 ;
30900 ;
31000 CNT01     CP      0AH     ;IS IT DOWN ARR?
31100           JP      Z,DNAR01
31200           CP      09H     ;IS IT FWD ARROW?
31300           JP      Z,FWAR01
31400           CP      0DH     ;IS IT CARRIAGE RETURN?
31500           JP      Z,DNAR01
31600           CP      08H     ;IS IT BACK SPACE?
31700           JP      Z,BKAR01
31800           CP      1FH     ;IS IT CLEAR KEY?
31900           JP      Z,CLER01
32000           CP      1BH     ;SHIFT UP ARROW?
32100           JP      Z,OPTION
32200           JP      LOOP08
32300 ;
32400 ;
32500 CLEAR     CALL    REPLCE
32600           CALL    SC2MEM  ;SAVE FORM IN MEMORY
32700           LD      HL,SCREEN
32800           LD      (IX),00H  ;INITIALISE NO. OF FLDS =0
32900           PUSH    IX
33000           POP     DE      ;DE NOW HAS TABLE START
```

```
33100          DEC     HL
33200 LOOP04   INC     HL        ;ADVANCE SCREEN POINTER
33300          CALL    TSTEND    ;AT END OF SCREEN?
33400          JP      P,OPTION  ;IF SO, DONE, SO RETURN
33500          LD      A,(HL)    ;GET CHARACTER ON SCREEN
33600          CP      5FH       ;IS IT CURSOR CHAR?
33700          JP      Z,ENTRY   ;IF SO, NEW FIELD
33800          JR      LOOP04
33900 ENTRY    INC     (IX)      ;INCREASE TOTL NO. FIELDS
34000          INC     DE
34100          LD      A,L       ;SAVE ADDRESS OF THIS FIELD
34200          LD      (DE),A    ; IN TABLE
34300          INC     DE        ;
34400          LD      A,H       ;
34500          LD      (DE),A
34600          INC     DE
34700          LD      A,01H     ;INITIALISE LENGTH OF FIELD =1
34800          LD      (DE),A
34900 LOOP05   LD      A,(DE)
35000          INC     A
35100          CP      240D      ;MAX FIELD LENGTH
35200          JP      Z,LOOP04  ;IF GREATER, CUT OFF.
35300          LD      (DE),A    ;LENGTH NOW INCREMENTED
35400          INC     HL
35500          CALL    TSTEND    ;AT END OF SCREEN?
35600          JP      P,OPTION
35700          LD      A,(HL)
35800          CP      5FH       ;ANOTHER CURSOR CHAR?
35900          JP      NZ,LOOP04 ;IF NOT, END OF FIELD
36000          JP      LOOP05
36100 ;
36200 TSTEND   LD      A,H
36300          CP      40H       ;HL GREATER THAN 3FFFH?
36400          RET
36500 ;
36600 ;
36700 VALACC   LD      A,(41E8H); GET CHAR IN BUFFER
36800          SUB     30H; CHANGE ASCII TO NO.
36900          RET
37000 ;
37100 GETKEY   PUSH    BC
37105          LD      BC,AMTDLY; AMOUNT OF DELAY FOR KEYBOARD
37115          CALL    DELAY   ;  DEBOUNCE.
37200          LD      A,(HL)    ;GET CHAR FROM SCREEN
37300          LD      (IY),A    ;SAVE IN IY
37400 LP       LD      (HL),143D  ;PUT BLOCK CURSOR ON SCREEN
37500          LD      B,00H
37600 LOOP25   CALL    GETCHR    ;GET KEY DEPRESSED
37700          CP      00H       ;IF ZERO, NO KEY DEPRESSED
37800          JP      NZ,SKIP25
38300          DJNZ    LOOP25    ;REPEAT 256 TIMES
38400          LD      A,(IY)    ;RETURN CHARACTER.
38500          LD      (HL),A    ;DISPLAY IT
38600          LD      B,00H
38700 LOOP26   CALL    GETCHR    ;GET KEY DEPRESSED
38800          CP      00H
38900          JP      NZ,SKIP25
39400          DJNZ    LOOP26   ;REPEAT 256 TIMES (FOR BLINK)
39500          JP      LP
39600 SKIP25   POP     BC
39700          RET     ;WE'VE GOT A CHARACTER, SO RETURN
39800 ;
39805 CLER01   CALL    REPLCE
39900          LD      B,01H
40000 LOOP13   LD      A,08H
40100          LD      (IY+4),A ;INITIALISE LENGTH OF DATA LINE
40200          LD      HL,(ENDPRG) ;CURRENT END OF PROGRAM PTR
40300          LD      DE,(40B1H)  ;MEMORY SIZE
40400          XOR     A
40500          PUSH    HL
40600          SBC     HL,DE   ;ARE WE OUT OF MEMORY?
40700          JP      P,OMERR ;OUT OF MEMORY ERROR IF SO.
40800          POP     HL
40900          LD      E,(IY+2)          ;GET NEW LINE NO.
41000          LD      D,(IY+3)          ; FOR DATA LINE
41100          INC     DE
41200          INC     DE
41300          INC     DE
41400          INC     DE
41500          INC     DE        ;INCREMENT IT BY 5
41600          LD      (IY+2),E
41700          LD      (IY+3),D   ;STORE IT FOR NEXT TIME
41800          LD      (HL),E     ;PUT IT IN DATA LINE
41900          INC     HL
42000          LD      (HL),D     ;LINE NO. NOW IN PLACE
42100          INC     HL
42200          LD      (HL),136D         ;DATA TOKEN
42300          INC     HL
42400          LD      (HL),32           ;SPACE
42500          INC     HL
42600          PUSH    HL
42700 LOOP15   CALL    GETFLD   ;GET ADRESS OF FIELD.
42800          POP     DE       ;DE NOW POINTS TO DATA LINE,
42900          LD      C,(IY+1)          ;LENGTH OF CURRENT FIELD
43000          LD      A,(IY+4)          ;LENGTH OF LINE SO FAR
43100          ADD     A,C
43200          JP      C,ENDLNE          ;NEED NEW DATA LINE
43300 ;IF WE GET HERE, THERE'S ENOUGH ROOM IN LINE
43400 LOOP14   LD      A,(HL)   ;GET CHAR FROM SCREEN
43500          CP      22H      ;QUOTES?
43600          JP      NZ,SKIP28
43700          LD      A,5BH    ;UP ARROW INSTEAD
43800          JP      SKIP31
43900 SKIP28   CP      2CH      ;COMMA?
44000          JP      NZ,SKIP29
44100          LD      A,5CH    ;DOWN ARROW INSTEAD
44200          JP      SKIP31
44300 SKIP29   CP      3AH      ;COLON?
44400          JP      NZ,SKIP30
44500          LD      A,5EH    ;FORWARD ARROW INSTEAD
```

```
44600          JP      SKIP31
44700 SKIP30   CP      5FH      ;ENTRY FIELD (CURSOR)?
44800          JP      Z,SKIP32 ; IF SO, NO MORE DATA IN FIELD
44805 SKIP31   DEC     C     ;C HAS NO. OF PLACES IN FIELD
44810          JP      Z,SKIP32 ;IF ZERO, AT END OF FIELD
44900          LD      (DE),A   ;LOAD CHAR INTO DATA LINE
45000          INC     DE
45100          INC     HL
45200          INC     (IY+4)
45205          JP      LOOP14   ;DO IT AGAIN
45500 SKIP32   LD      A,2CH    ;INSERT COMMA
45600          LD      (DE),A
45700          INC     DE
45800          INC     HL
45900          INC     (IY+4)
46000          INC     B        ;NEXT FIELD
46100          PUSH    DE
46200          LD      A,(IX)   ;ARE WE AT END OF
46300          CP      B        ; FIELDS?
46400          JP      P,LOOP15  ;IF NOT, GET NEXT
46500          POP     DE
46600 ENDLNE   XOR     A
46700          DEC     DE    ;END OF LINE ZERO OVER LAST COMMA
46800          LD      (DE),A
46900          INC     DE
47000          LD      (DE),A   ;END OF PROGRAM ZEROS
47100          INC     DE
47200          LD      (DE),A
47300          INC     DE       ;NOW AT NEW VARIABLE START
47400          LD      HL,(ENDPRG)  ;GET OLD VARIABLE START
47500          PUSH    HL       ;SAVE THIS POSITION
47600          LD      HL,ENDPRG
47700          LD      (HL),E
47800          INC     HL
47900          LD      (HL),D   ;RESET ENDPRG TO NEW POSITION
48000          POP     HL       ;OLD VAR START IS SPOT
48100          DEC     HL       ; FOR NEW LINE POINTER
48200          DEC     HL       ;
48300          DEC     DE
48400          DEC     DE
48500          LD      (HL),E   ;RESET LINE POINTER
48600          INC     HL
48700          LD      (HL),D
48800          LD      A,(IX)   ;TOTAL NO OF FIELDS
48900          CP      B        ;ARE WE FINISHED?
49000          JP      P,LOOP13  ;NO, SO CONTINUE
49100          JP      FILL
49200 ;
49300 YTAB     DEFS    10H      ;TABLE FOR ODD VARIABLES
49400 XTAB     DEFW    0000H    ;LOOKUP TABLE FOR ENTRY FIELDS
49500          DEFW    0000H
49600          DEFS    252D
49700 ;
49800 LAST     NOP
49900          END     BEGIN
```

```
433F:  00 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
434F:  20 4D 41 49 4C 49 4E 47 20 4C 49 53 54 20 44 45
435F:  4D 4F 4E 53 54 52 41 54 49 4F 4E 20 20 20 20 20
436F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
437F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
438F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
439F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
43AF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
43BF:  20 53 55 52 4E 41 4D 45 20 5F 5F 5F 5F 5F 5F 5F
43CF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 49
43DF:  4E 49 54 49 41 4C 53 20 5F 20 5F 20 20 20 20 20
43EF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
43FF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
440F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
441F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
442F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
443F:  20 53 54 52 45 45 54 2F 52 4F 41 44 20 5F 5F 5F
444F:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F
445F:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 20 20 20
446F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
447F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
448F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
449F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
44AF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
44BF:  20 54 4F 57 4E 2F 53 55 42 55 52 42 20 5F 5F 5F
44CF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F
44DF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 20 20 20
44EF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
44FF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
450F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
451F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
452F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
453F:  20 53 54 41 54 45 20 20 20 20 20 20 20 5F 5F 5F
454F:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F
455F:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 20 20 20
456F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
457F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
458F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
459F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
45AF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
45BF:  20 43 4F 55 4E 54 52 59 20 20 20 20 20 5F 5F 5F
45CF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F
45DF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 20 20 20
45EF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
45FF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
460F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
461F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
462F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
463F:  20 50 4F 53 54 20 43 4F 44 45 20 20 20 5F 5F 5F
464F:  5F 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
465F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
466F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
467F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
468F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
```

```
469F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
46AF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
46BF:  20 50 48 4F 4E 45 20 20 20 20 20 20 20 5F 5F 5F
46CF:  5F 5F 5F 5F 5F 5F 5F 5F 5F 20 20 20 20 20 20 20
46DF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
46EF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
46FF:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
470F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
471F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
472F:  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
473F:  20 20 20 20 20 20 20 20 2A 20 46 20 4F 20 52 20
474F:  4D 20 41 20 54 20 49 20 4F 20 4E 20 2A 0A 43 4F
475F:  50 59 52 49 47 48 54 20 31 39 38 32 20 44 41 56
476F:  49 44 20 52 2E 20 47 52 49 47 47 0A 0A 4F 50 54
477F:  49 4F 4E 53 0A 0A 31 2E 20 4D 4F 44 49 46 59 20
478F:  45 58 49 53 54 49 4E 47 20 46 4F 52 4D 0A 32 2E
479F:  20 43 52 45 41 54 45 20 4E 45 57 20 46 4F 52 4D
47AF:  0A 33 2E 20 53 54 4F 52 45 20 4E 45 57 4C 59 2D
47BF:  43 52 45 41 54 45 44 20 46 4F 52 4D 0A 34 2E 20
47CF:  46 49 4C 4C 20 45 58 49 53 54 49 4E 47 20 46 4F
47DF:  52 4D 53 0A 35 2E 20 52 45 54 55 52 4E 20 54 4F
47EF:  20 42 41 53 49 43 0A 00 21 3B 48 22 7A 41 21 81
47FF:  4C AF 77 23 77 22 A4 40 23 77 23 22 F9 40 22 FB
480F:  40 22 FD 40 FD 21 71 4B 3E 7D FD 77 03 AF FD 77
481F:  02 C3 CC 06 46 4F 52 4D 41 54 49 53 20 52 45 43
482F:  4F 52 44 45 52 20 52 45 41 44 59 00 DD 21 81 4B
483F:  FD 21 71 4B CD C9 01 21 40 47 CD A7 28 CD B3 1B
484F:  D7 CD 93 4A FE 01 CA 87 48 FE 02 CA 8D 48 FE 03
485F:  CA 24 49 FE 04 CA A5 49 FE 05 CA CC 06 C3 3B 48
486F:  01 00 04 21 00 3C 11 40 43 ED B0 C9 01 00 04 21
487F:  40 43 11 00 3C ED B0 C9 CD 7B 48 C3 90 48 CD C9
488F:  01 21 00 3C CD 99 4A FE 20 FA BC 48 FE 5B CA AD
489F:  48 FE 40 CA D8 48 77 23 CD 03 49 C3 93 48 CD DD
48AF:  48 11 40 00 AF ED 52 CD 0B 49 C3 93 48 FE 08 CA
48BF:  E2 48 FE 0A CA EC 48 FE 09 CA F9 48 FE 0D CA 13
48CF:  49 FE 1B CA 4A 4A C3 93 48 3E 5F C3 A5 48 FD 7E
48DF:  00 77 C9 CD DD 48 2B CD 0B 49 C3 93 48 CD DD 48
48EF:  11 40 00 19 CD 03 49 C3 93 48 CD DD 48 23 CD 03
48FF:  49 C3 93 48 7C FE 40 F8 21 FF 3F C9 7C FE 3C F0
490F:  21 00 3C C9 CD DD 48 11 40 00 19 7D E6 C0 6F CD
491F:  03 49 C3 93 48 CD C9 01 21 29 48 CD A7 28 CD B3
492F:  1B AF CD 12 02 21 81 4C 11 3F 43 AF ED 52 23 CD
493F:  87 02 3E 55 CD 64 02 06 06 E5 21 23 48 7E CD 64
494F:  02 23 10 F9 E1 25 FA 66 49 3E 3C CD 64 02 AF CD
495F:  64 02 CD 8C 49 18 EE AF BD 28 0C 3E 3C CD 64 02
496F:  7D CD 64 02 CD 8C 49 3E 78 CD 64 02 01 F7 47 79
497F:  CD 64 02 78 CD 64 02 CD F8 01 C3 3B 48 47 7B CD
498F:  64 02 7A CD 64 02 83 4F 1A CD 64 02 81 4F 13 10
499F:  F7 79 CD 64 02 C9 CD 7B 48 06 01 CD 0D 4A 7E FD
49AF:  77 00 CD 99 4A FE 20 FA 29 4A FE 5B CA D0 49 FE
49BF:  60 CA C7 4A 77 0C 23 FD 7E 01 B9 CA E3 49 C3 B1
49CF:  49 CD DD 48 05 C2 DA 49 DD 46 00 CD 0D 4A C3 AD
49DF:  49 CD DD 48 04 DD 7E 00 B8 F2 ED 49 06 01 CD 0D
49EF:  4A C3 AD 49 CD DD 48 2B 0D C2 B1 49 C3 D3 49 CD
49FF:  DD 48 23 0C FD 7E 01 B9 FA E0 49 C3 B1 49 0E 01
```

```
4A0F:  DD E5 D1 13 C5 05 CA 1D 4A 13 13 13 18 F7 1A 6F
4A1F:  13 1A 67 13 1A FD 77 01 C1 C9 FE 0A CA E0 49 FE
4A2F:  09 CA FE 49 FE 0D CA E0 49 FE 08 CA F3 49 FE 1F
4A3F:  CA C7 4A FE 1B CA 3B 48 C3 B1 49 CD DD 48 CD 6F
4A4F:  48 21 00 3C DD 36 00 00 DD E5 D1 2B 23 CD 8F 4A
4A5F:  F2 3B 48 7E FE 5F CA 6A 4A 18 F1 DD 34 00 13 7D
4A6F:  12 13 7C 12 13 3E 01 12 1A 3C FE F0 CA 5B 4A 12
4A7F:  23 CD 8F 4A F2 3B 48 7E FE 5F C2 5B 4A C3 77 4A
4A8F:  7C FE 40 C9 3A E8 41 D6 30 C9 C5 01 00 13 CD 60
4A9F:  00 7E FD 77 00 36 8F 06 00 CD 2B 00 FE 00 C2 C5
4AAF:  4A 10 F6 FD 7E 00 77 06 00 CD 2B 00 FE 00 C2 C5
4ABF:  4A 10 F6 C3 A4 4A C1 C9 CD DD 48 06 01 3E 08 FD
4ACF:  77 04 2A F9 40 ED 5B B1 40 AF E5 ED 52 F2 A7 19
4ADF:  E1 FD 5E 02 FD 56 03 13 13 13 13 13 FD 73 02 FD
4AEF:  72 03 73 23 72 23 36 88 23 36 20 23 E5 CD 0D 4A
4AFF:  D1 FD 4E 01 FD 7E 04 81 DA 4D 4B 7E FE 22 C2 15
4B0F:  4B 3E 5B C3 2E 4B FE 2C C2 1F 4B 3E 5C C3 2E 4B
4B1F:  FE 3A C2 29 4B 3E 5E C3 2E 4B FE 5F CA 3B 4B 0D
4B2F:  CA 3B 4B 12 13 23 FD 34 04 C3 0A 4B 3E 2C 12 13
4B3F:  23 FD 34 04 04 D5 DD 7E 00 B8 F2 FC 4A D1 AF 1B
4B4F:  12 13 12 13 12 13 2A F9 40 E5 21 F9 40 73 23 72
4B5F:  E1 2B 2B 1B 1B 73 23 72 DD 7E 00 B8 F2 CC 4A C3
4B6F:  A5 49 20 08 0F 7D 20 52 45 43 4F 52 44 45 52 50
4B7F:  46 0B 09 88 3C 15 A7 3C 02 A9 3C 02 0C 3D 1F 8C
4B8F:  3D 1F 0C 3E 1F 8C 3E 1F 0C 3F 05 8C 3F 0D 53 45
4B9F:  54 20 54 4F 20 42 4F 55 4E 44 41 52 49 45 53 20
4BAF:  4F 46 20 50 52 4F 47 52 41 4D 18 47 06 09 58 4F
4BBF:  52 09 41 7C 47 1F 09 53 42 43 09 48 4C 2C 44 45
4BCF:  09 3B 4E 4F 20 4F 46 20 42 59 54 45 53 20 54 4F
4BDF:  20 4D 4F 56 45 E0 47 07 09 49 4E 43 09 48 4C 44
4BEF:  48 26 09 43 41 4C 4C 09 4C 45 44 45 52 09 3B 57
4BFF:  52 49 54 45 20 4C 45 41 44 45 52 20 26 20 53 59
4C0F:  4E 43 48 20 42 59 54 45 A8 48 09 09 4C 44 09 41
4C1F:  2C 35 35 48 0C 49 0B 09 43 41 4C 4C 09 57 52 49
4C2F:  54 45 70 49 09 09 4C 44 09 42 2C 30 36 48 D4 49
4C3F:  08 09 50 55 53 48 09 48 4C 38 4A 25 09 4C 44 09
4C4F:  48 4C 2C 4E 41 4D 45 09 3B 50 52 4F 47 52 41 4D
4C5F:  20 4E 41 4D 45 20 49 53 20 27 46 4F 52 4D 41 54
4C6F:  27 9C 4A 10 4C 4F 4F 50 34 30 09 4C 44 09 41 2C
4C7F:  28 48 00
```

****** L2/16K   ORDERING PRIORITIES ******

TRS-80/SYSTEM-80

```
10 'P R I O R I T I E S
20 'AFJ Bell  20/06/81
30 '49 Hyde Park Rd, TRARALGON, 3844
40 'Uses 5.3 KBytes including ULCBAS
50 CLS:IF PEEK(14312)<>63 THEN PRINT"Lineprinter not ready"
60 LPRINT CHR$(27);CHR$(66):POKE 16424,72:POKE 16425,0:LN=0
70 CLEAR 1000:DEFINT A-Z
```

```
80 DIM SD$(12,3),HEAD$(4),PU$(3),OPTN$(9),I(12)
90 FOR L=0 TO 4:READ HEAD$(L):NEXT  'Sets headings etc
100 DATA PRIORITIES,Raw data,by Importance,by Feasibility,by Pri
ority
110 FOR L=0 TO 3:READ PU$(L):NEXT
120 DATA "%                          %","%  %","%  %","%  %"
130 CLS  'Sets & displays options
140 FOR L=1 TO 9:READ OPTN$(L):NEXT
150 DATA Display Instructions,Input Data,Echo Data
160 DATA Add Line(s),Change Line
170 DATA Order by Importance,Order by Feasibility
180 DATA Order by Priority, End
190 CLS:PRINT:PRINT CHR$(23) TAB(12)"Options":PRINT
200 FOR L=1 TO 9:PRINT L;"   ";OPTN$(L):NEXT
210 PRINT:PRINT"What is your choice ?"
220 O$=INKEY$:IF O$="" GOTO 220
230 IF O$<"1" OR O$>"9" GOTO 220 ELSE OPTN=VAL(O$)
240 ON OPTN GOTO 250,310,450,470,480,580,610,640,670
250 CLS:PRINT:PRINT CHR$(23)  'Instructions
260 PRINT" This program will help to dec-";CHR$(13);"ide the rel
ative priorities of";CHR$(13);"various activities."
270 PRINT" Please enter data as prompted.";CHR$(13);"You must pr
ess ENTER if the";CHR$(13);"cursor is showing. If a mistake";CHR
$(13);"is made when entering data then";CHR$(13);"enter XXX as a
ctivity name. To"
280 PRINT"finish entering data enter ZZZ ";CHR$(13);"as activity
 name."
290 PRINT:PRINT"Press any key when ready"
300 R$=INKEY$:IF R$="" GOTO300 ELSE GOTO 190
310 CLS:PRINT CHR$(23)  'Accepts data
320 GOTO 980
330 N=N+1:I(N)=N:IF N>12 PRINT"No more space":FOR L=1 TO 500:NEX
T:GOTO 190
340 INPUT"Activity name ";A$:SD$(I(N),0)=LEFT$(A$,25)
350 IF SD$(I(N),0)="XXX" THEN N=N-1:PRINT"Rewrite entry":GOTO 34
0
360 IF SD$(I(N),0)="ZZZ" THEN N=N-1:GOTO 190
370 PRINT"Importance    ? ";
380 I$=INKEY$:IF I$="" GOTO 380
390 IF I$<"1" OR I$>"9" GOTO 380 ELSE PRINT I$:SD$(I(N),1)=I$
400 PRINT"Feasibility   ? ";
410 F$=INKEY$:IF F$="" GOTO 410
420 IF F$<"1" OR F$>"9" GOTO 410 ELSE PRINT F$:SD$(I(N),2)=F$
430 SD$(I(N),3)=STR$(VAL(SD$(I(N),1))*VAL(SD$(I(N),2)))
440 PRINT:GOTO 330
450 CLS  'Echoes data
460 H=1:GOSUB 680:GOSUB 850:GOTO 190
470 CLS:PRINT CHR$(23)"Add more line(s)":GOTO 330
480 CLS:PRINTCHR$(23)"Which line will you change";:INPUT C
490 INPUT"Activity name  ";SD$(I(C),0)
500 PRINT"Importance    ? ";
510 I$=INKEY$:IF I$="" GOTO 510
520 IF I$<"1" OR I$>"9" GOTO 510 ELSE PRINT I$:SD$(I(C),1)=I$
530 PRINT"Feasibility   ? ";
540 F$=INKEY$:IF F$="" GOTO 540
550 IF F$<"1" OR F$>"9" GOTO 540 ELSE PRINT F$:SD$(I(C),2)=F$
560 SD$(I(C),3)=STR$(VAL(SD$(I(C),1))*VAL(SD$(I(C),2)))
570 GOTO 190
580 T=1:H=2
590 PRINT@ 960,"Ordering by importance..";
600 GOSUB 770:GOTO 190
610 T=2:H=3
620 PRINT@ 960,"Ordering by feasibility...";
630 GOSUB 770:GOTO 190
640 T=3:H=4
650 PRINT@ 960,"Ordering by priority...";
660 GOSUB 770:GOTO 190
670 CLS:PRINT"Goodbye":GOSUB 950:END
680 CLS:TITLE$=HEAD$(0)+"  "+HEAD$(H)  'Display subroutine
690 PRINT TAB(32-LEN(TITLE$)/2) TITLE$:PRINT
700 PRINT"    Activity";TAB(35)"Imp Fsb Pri"
710 FOR L=1 TO N
720  PRINT USING PU$(1);STR$(L);
730  FOR J=0 TO 3
740   PRINT USING PU$(J);SD$(I(L),J);
750  NEXT J:PRINT
760 NEXT L:RETURN
770 FOR K=1 TO N-1  'Secondary index Bubble "Sort" & Display
    subroutine
780  F=0
790  FOR J=1 TO N-K
800   IF VAL(SD$(I(J),T)) >= VAL(SD$(I(J+1),T)) GOTO 820
810   F=1:X=I(J):I(J)=I(J+1):I(J+1)=X
820  NEXT J:IF F=0 GOTO 840
830 NEXT K
840 GOSUB 680:GOSUB 850:RETURN
850 PRINT@ 960,"Press C to copy, O for options";  'Screenprint s
ubroutine
860 R$=INKEY$:IF R$="" GOTO 860
870 IF R$="O" GOTO 190 ELSE IF R$="C" GOTO 880 ELSE GOTO 860
880 V=15360
890 FOR R=0 TO 14
900  FOR C=0 TO 63
910   LPRINT CHR$(PEEK(V+64*R+C));
920 NEXT C,R:LPRINT" ":LN=LN+16:POKE 16425,LN
930 IF LN=>64 GOSUB 950
940 RETURN
950 IF PEEK(16425)<70 THEN LPRINT" ":GOTO 950  'Pagination subro
utine
960 CLS:PRINT:PRINT:PRINT"Please tear off page, then press any k
ey"
970 R$=INKEY$:IF R$="" GOTO 970 ELSE POKE 16425,1:LN=0:RETURN
980 CLS:INPUT"N = ";N  'Test data subroutine
990 FOR L=1 TO N
1000  I(L)=L
1010  N0=RND(26):SD$(I(L),0)=STRING$(15,CHR$(N0+64))
1020  N1=RND(9) :SD$(I(L),1)=STR$(N1)
1030  N2=RND(9) :SD$(I(L),2)=STR$(N2)
1040  SD$(I(L),3)=STR$(N1*N2)
1050 NEXT:GOTO 190
```

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

Next month's issue will contain at least the following programs plus the usual features and articles. An (80) after a program title indicates that the program will be for TRS-80 Model 1/3 or System 80/Video Genie. A (CC) indicates that the program will be for the TRS-80 Colour Computer and (Peach) that the program is for the Hitachi Peach.

** HAMBURGER (80) L2/16K **

It is your job to make hamburgers. You have got some buns, eggs, salad and meat. The only trouble is that while you are trying to put the hamburgers together, a couple of aggressive sausages are trying to catch you. The bigger the hamburger, the bigger your score.

** XUSR SCREEN FILL SUBROUTINE (80) L2/4K **

This very short but powerful subroutine gives you a new function that you can use within your own programs. With this program you can insert X=USR(N) where N is a number from 1 to 255 and the screen will fill with that character at machine language speed.

** WORLD CHAMPIONSHIP BOXING (CC) **

If you ever felt like a punch-up with your computer, now's your chance. The computer controls one of the boxers and you control the other. If you don't want to get knocked out you better keep on the move - the computer will show no mercy!!

** OTHELLO (HP) **

This is the same game, (different program and author though), as the one that appeared last issue for the Colour Computer. The game is played on an 8 x 8 board and you must outflank your opponent to flip his playing pieces to your colour.

** HIGH RESOLUTION SCREEN SCORE SUBROUTINE (CC) **

This subroutine can be added to any basic program that requires a score display on a high resolution screen. The size of the score numbers and their position can easily be changed to suit your needs - all that has to be done to use the subroutine is to use the variable SC to contain your score and then call the subroutine.

** SYSTEM TAPE MAKER (80) L2/16K **

Just for the Level 2 users, this utility gives your machine the ability to save any block of memory to tape, either from a BASIC program or from the command mode. All you have to do is type - SAVE name addr1 addr2 addr3. Where name is a six digit name, addr1 is the start address, addr2 is the end address and addr3 is the optional entry point.

** BLOCK GAME (HP) **

This is a two player game in which each player controls a graphic line within a confined area. As each player's line moves, it gets longer and longer. You must try to force your opponent to run into the border - himself or you. The first player to hit something is the loser.

APPLICATION FOR PUBLICATION OF A PROGRAM IN MICRO-80

Date ....

To MICRO-80
SOFTWARE DEPT.,
P.O. BOX 145,
MORPHETT VALE, S.A. 5162

Please consider the enclosed program for publication in MICRO-80.

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 applicable software listed each month, on cassette. For machine language programs copies of both the source and object file are provided. All programs are recorded twice. Level 1 programs can only be loaded into a Level 2 machine if the 'Level 1 in Level 2' program from the MICRO-80 Software Library - Vol. 1 is loaded first.

Note: System 80/Video Genie 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 applicable programs which can be executed from disk. Level 1 disk programs are saved in NEWDOS format. Users require the Level I/CMD utility supplied with NEWDOS+ or NEWDOS 80 version 1.0 to run them.

				APPROX. START POSITION		
SIDE 1	TYPE	I.D.	DISK FILESPEC	CTR-41	CTR-80	SYSTEM 80
PRIORITIES	L2/16K	P	PRIOR/BAS	18	10	5
"	"	"	"	63	35	17
DESERT CHASE	L2/16K	D	DESERT/BAS	105	58	30
"	"	"	"	184	103	58
FORMATION	SYSTEM	FORMAT	FRMATION/CMD	257	144	84
"	"	"	"	281	157	92
FORMATION	EDTASM	FORMAT	FRMATION/EDT	304	170	103
SIDE 2						
FORMATION	EDTASM	FORMAT	FRMATION/EDT	18	10	5
OTHELLO	COCO	O	-	170	95	-
"	"	"	-	193	108	-
TOWER OF HANOI	COCO	T	-	215	120	-
"	"	"	-	237	133	-
TOWER OF HANOI	HITACHI	TOWERS	-	261	146	-
"	"	"	-	323	181	-
REGISTER DISPLAY	HITACHI	REG/BAS	-	330	213	-
"	"	"	-	391	219	-

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 mth. subs. to MICRO-80, plus the cassette edition

.... 12 mth. subs. to MICRO-80, plus the disc 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

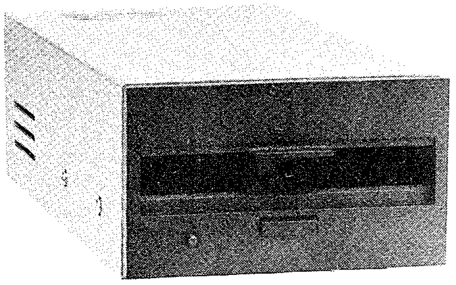
Signature .... Exp. End ....

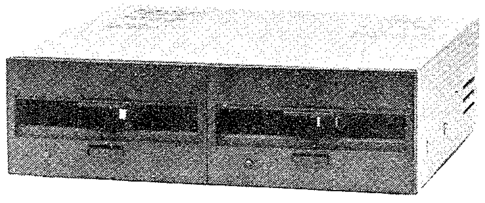
NAME ....

ADDRESS ....

.... Postcode ....

SAVE A PACKET ON MICRO-80's DISK DRIVE PACKAGES FOR TRS-80 MODEL 1 AND SYSTEM 80 MICROCOMPUTERS





SINGLE DRIVE PACKAGE *from* ... $499 **DUAL DRIVE PACKAGE** *from* ... $874

Bigger volume means lower cost price, which we are passing on to you. Avoid the annoying bundle of cables, wires and separate boxes. MICRO-80 is now offering our well-proven MPI disk drives in attractive, self-contained single or dual-drive cabinets complete with internal power supply. Our drive Ø and dual-drive packages also include the appropriate version of DOSPLUS and dual-drive cable.

The best news of all is the specially reduced package prices ...
SAVE $23 — $107 over our already low prices!

Choose the appropriate system from the table below:

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price	* Saving
DRIVE Ø						
1 x MPI B51	40	1	100K	3.4	$499	$137.95
1 x MPI B52	40	2	200K	3.4	$639	$97.95
1 x MPI B92	80	2	400K	3.4	$799	$107.95
DRIVE 1						
1 x MPI B51	40	1	100K	—	$415	$33.00
1 x MPI B52	40	2	200K	—	$525	$23.00
1 x MPI B92	80	2	400K	—	$695	$23.00

*Represents the saving compared with buying all the items included in the package separately

•Drive Ø package includes one bare disk drive, self-contained single-drive cabinet/power supply as illustrated, two drive cable and the version of DOSPLUS indicated.

•Drive 1 package includes one bare disk drive and self-contained single-drive cabinet/power supply as illustrated.

If it's a dual-drive system you need, then take advantage of our dual-drive package and
SAVE a further $40 on the price of two single-drive packages ...

DRIVE TYPE	No. of Tracks	No. of Heads	Capacity	Dosplus Version	Price
2 x MPI B51	40 ea	1 ea	2 x 100K	3.4	$874
2 x MPI B52	40 ea	2 ea	2 x 200K	3.4	$1125
2 x MPI B92	80 ea	2 ea	2 x 400K	3.4	$1454

Dual-drive package includes two bare disk drives, self-contained dual-drive cabinet/power supply as illustrated, two drive cables and the version of Dosplus indicated.

NOTE: All 40 track drives are completely compatible with 35 track operating systems such as TRSDOS. DOSPLUS allows you to realise an additional 14% capacity compared with TRSDOS. Under DOSPLUS 3.4, 80 track drives can read 35/40 track diskettes.

All disk drive components are still available separately:

BARE DRIVES — MPI drives offer the fastest track-to-track access time (5 milliseconds) available. All drives are capable of operating in double density for 80% greater storage capacity.

	Price	Freight
MPI B51 40 track, single-head, 100K	$349	$5.00
MPI B52 40 track, dual-head, 200K	$449	$5.00
MPI B92 80 track, dual-head, 400K	$619	$5.00
Separate, dual-drive power supply	$85	$8.00

	Price	
Self-contained, single drive cabinet/power supply	$99	$5.00
Self-contained, dual-drive cabinet/power supply	$135	$5.00
Two drive cable	$39	$2.00
Four drive cable	$49	$2.00
DOSPLUS 3.4	$149.95	$2.00

Prices are FOB Adelaide. Add $5.00 freight for single drive package, $10.00 for dual-drive package. Prices are in Australian dollars. Freight is road freight anywhere in Australia.

All items carry a 90-day parts and labour warranty. Repairs to be carried out in our Adelaide workshops.

LEVEL 2 ROM

ASSEMBLY LANGUAGE TOOLKIT

by Edwin Paay

FOR TRS-80 MODEL 1, MODEL 3 AND SYSTEM 80/VIDEO GENIE

This is a new package consisting of two invaluable components:

- **A ROM REFERENCE** Manual which catalogues, describes and cross-references the useful and usable ROM routines which you can incorporate into your own machine language or BASIC programs.
- **DBUG**, a machine language disassembling debugging program to speed up the development of your own machine language programs. DBUG is distributed on a cassette and may used from disk or cassette.

Part 1 of the ROM REFERENCE manual gives detailed explanations of the processes used for arithmetical calculations, logical operations, data movements etc. It also describes the various formats used for BASIC, System and Editor/Assembly tapes. There is a special section devoted to those additional routines in the TRS-80 Model 3 ROM. This is the first time this information has been made available, anywhere. Differences between the System 80/Video Genie are also described. Part 1 is organised into subject specific tables so that you can quickly locate all the routines to carry out a given function and then choose the one which meets your requirements.

Part 2 gives detailed information about each of the routines in the order in which they appear in the ROM. It describes their functions, explains how to use them in your own machine language programs and notes the effect of each on the various Z80 registers.

Part 2 also details the contents of system RAM and shows you how to intercept BASIC routines. With this knowledge, you can add your own commands to BASIC, for instance, or position BASIC programs in high memory — the only restriction is your own imaginationl

The Appendices contain sample programmes which show you how you can use the ROM routines to speed up your machine language programs and reduce the amount of code you need to write.

DBUG: Eddy Paay was not satisfied with any of the commercially available debugging programs, so he developed his own. DBUG: allows you to single-step through your program; has 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; displays/edits memory in Hex or ASCII; allows Register editing; has the ability to read and write System tapes and all this on the bottom 3 lines of your screen, thus freeing the rest of the screen for program displays. Four versions of DBUG are included in the package to cope with different memory sizes.

The best news of all Is the prlce. The complete Level 2 ROM ASSEMBLY LANGUAGE TOOLKIT Is only:

— Aus. $29.95 + $2.00 p&p
— UK £18.00 + £1.00 p&p

SPECIAL OFFER TO OWNERS OF THE LEVEL II ROM REFERENCE MANUAL ...
UPGRADE TO THIS ASSEMBLY LANGUAGE TOOKIT FOR ONLY $19.95!
Send back your orlglnal Level II ROM Reference Manual plus a cheque, money order or Bankcard authorlsatlon for $19.95 plus $2.00 p&p and we wlll send you the new ASSEMBLY LANGUAGE TOOLKIT