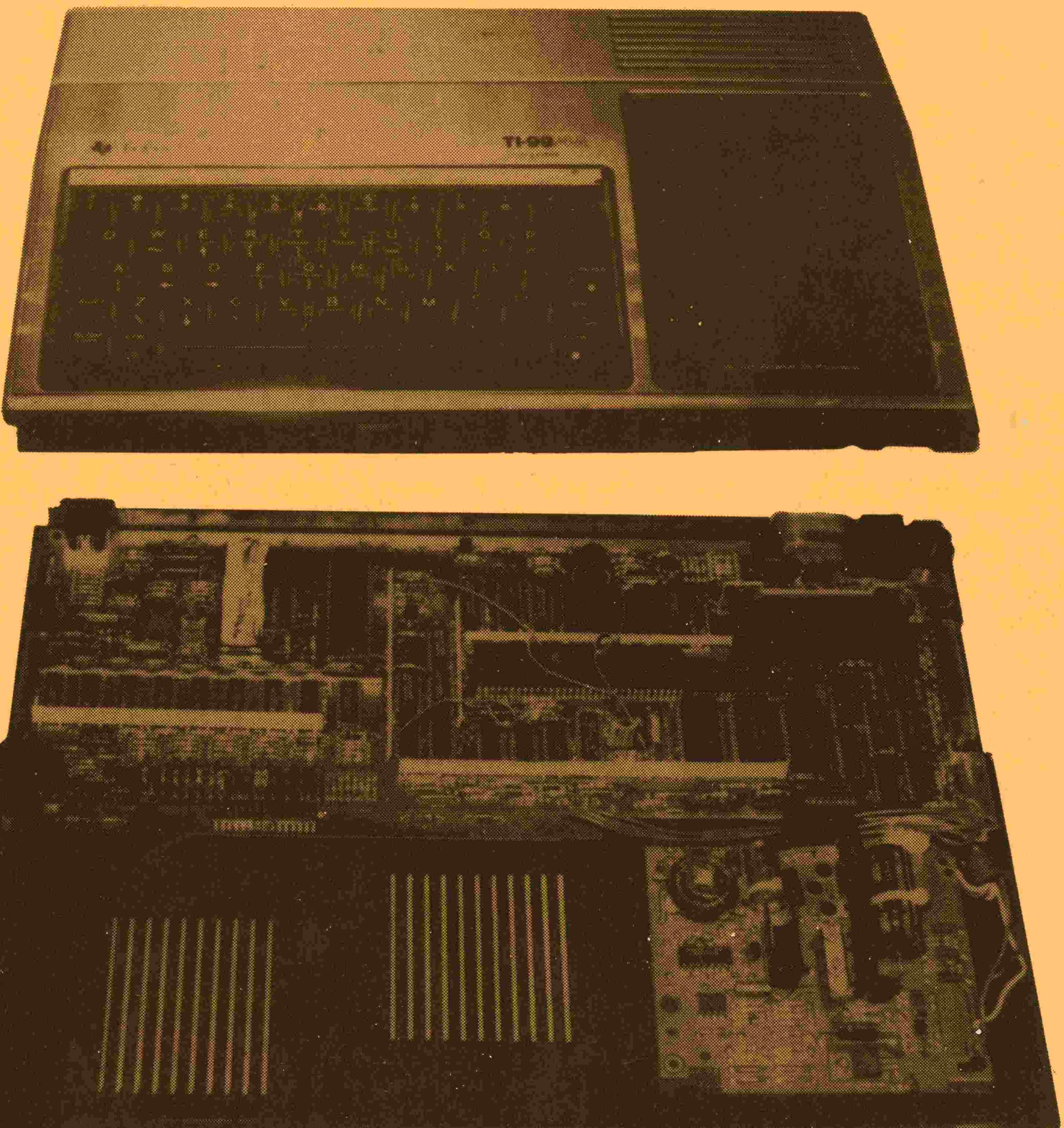
TI*MES



Issue 21

T.I.COMMITTEE MEMBERS 1988

Area of Responsibility

CONTENTS

TI99/4A USERS GROUP(U.K)

MINUTES OF A.G.M. 1988

Held at DERBY 28th May.

Chairman. Clive Scally

The Chairman opened the meeting, expressing his thanks to the Commiteee and all members who were in attendance.

He noted with regret that three stalwarts supporters of the T.I had been lost to the Group in the last few months - Ian Martin, John Roe and Ron Johnson, and felt that it was appropriate that they be remembered at this meeting, and their names be recorded in the Minutes.
A disc of Ron's programmes is available from the library.

AGENDA

1. Minutes of Last A.G.M.

Were approved by the meeting nem.con.

2. Chairman'Report.

The Chairman reported that the membership level was stable, and thanked Committee members and other parties for the work they had done in the past year. He drew attention to his report in the last issue of TI*MES, which made repetition now unnecessary.

He reported that the CINDY Group was now showing support, and emphasised the need for a continuing National Group, pointing to the very strong interest in the TI which was still very evident.

Members responsible for the conduct of the various library facilities had each, with one exception, been in office for only a short period and are only now moving into full operation.
The exception was Stephen Shaw, who ran the Disc Library, and had sent in an excellent report indicating a healthy surplus enabling him to purchase new material as it became available from overseas.

3. Treasurer's Report.

The Chairman reported that the Treasurer during 1987/8 had indicated that he was not seeking re-election. He reported a balance of £938.00 at the bank. He had the relevant literature and would pass this on to the new Treasurer when appointed.
He was happy to report that the financial position at the end of the first year was healthy.

4. Secretary's Report.
The Secretary reported that in spite of problems arising due to communication difficulty with such a widespread committee and postal delays, he was happy to report that the TIUG(UK) was now well prepared for its second successful year. He hoped to propose to the Committee minor alterations to the Constitution to avoid or at least minimise the delays that occurred occasionally last year.
He suggested that the Constitution, which had served the Group (pro tem) for twelve months now, should be legitimised by formal acceptance.

The Chairman moved " That the Constitution of the TIUG(UK) as circulated be formally accepted" This was put to the meeting and accepted nem.con.

5. Election of Committee Members and Officers.

The following were duly elected:-

Chairman.	Clive Scally.
General Secretary.	Jim Ballinger.
Membership Secretary,Communications.	Peter Walker.
Treasurer.	Alan Rutherford.
Cassette Librarian.	Tim Anderson.
Editor.Publications.	Mrs.Christina Mehew.
TI*MES Print and distribution.	Alan Bailey.
Disc Librarian,Rambles,International Journals.	Stephen Shaw.
Books and Publications Librarian.	Bryan Cloud.
Programming Languages.	Geoffrey Coan.

Hardware,D.I.Y.and Hardware Librarian.	..Mike Goddard.
Module Librarian.	.......................Edward Shaw.
Publicity Officer. *	...................Mrs.Christine Bennett

* New appointment to offset overload on Membership Secretary.

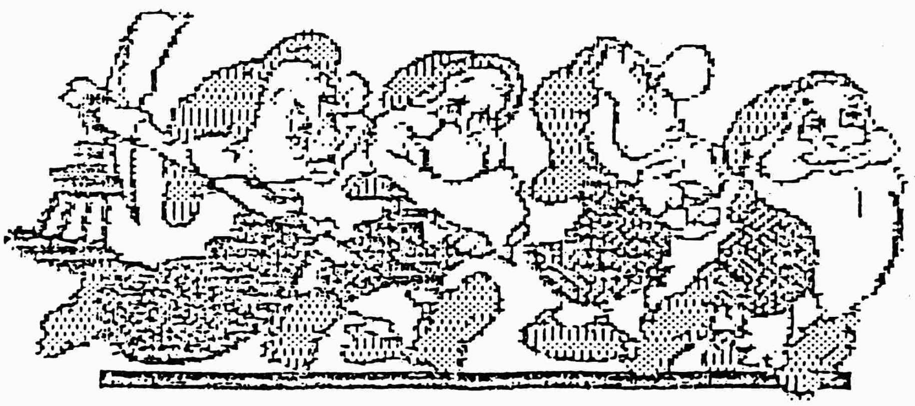
6. Any other business.

Maurice Rymill suggested that letters should be sent to machine specific magazines to find 'dual users' using TI/99's . Mike Goddard agreed to liaise with other groups to action this.

Peter Walker requested that more members write articles for the magazine (Programmes, comments etc) More articles were required for the unexpanded machine owner particularly. The Chairman asked members to express their own thoughts about this. Ross Bennett suggested that users should be urged to expand their system. Mike Goddard pointed out that 'home brew' projects can become very involved. A member pointed out that a large number of suitable programmes was available to members from the Cassettee Library.

Maurice Rymill congratulated those responsible for the layout, design and production of TI*MES latest issue. This met with general approval, though there was some feeling that a larger type-face would be an advantage. The Chairman asked for views about this from readers, should we go for larger print (which must involve less content), or keep the same style?

The Chairman thanked everyone for a very successful A.G.M and declared the meeting closed.



V.3

CONSTITUTION RULES OF THE TI99/4a USERS GROUP(UNITED KINGDOM)

1. The TIUGUK shall be known as"The TI99/4a Users Group(U.K)". (hereinafter called the Group)

2. The object of the Group will be to provide an open forum for discussion on T.I related computers, to provide and encourage opportunities for inter-trading between members, and to enable members to hear speakers on specialist subjects.

3. The Group shall be managed by a Committee to be elected at each Annual General Meeting(AGM) and shall include a Chairman, Vice-Chairman, Treasurer and Secretary (hereinafter called the Officers) and 10 other members, and shall have power, with the consent of any member, to co-opt him or her onto the Committee. Subject to termination of office by resignation or other reasons, the Committee shall remain in office until their successors are elected at the next A.G.M. The Committee shall have the power to fill any vacancy that may arise on the Comittee. The retiring Officers shall be eligible for re-election.

4. The annual subscription will be set by the Committee and subject to approval at the A.G.M. It will become due on joining , and must be renewed annually. The Committee may cancel the membership of any member whose subscription is 56 days or more in arrear.

5. Any member may bring one guest to any general meeting of the Group. No fee will be payable by the guest. No guest may attend as a guest at more than two meetings in any year without joining the Group.

6. The Committee shall have power to expel any member who shall offend against the rules of the Group, or if his conduct shall in the opinion of the Committee render him unfit for membership. Before any member may be expelled the Secretary shall give him/her seven days written notice of a meeting of the Committee, and must inform him/her of the complaints made against that member. No member shall be expelled without first having the opportunity of appearing before the Committee to answer complaints made against him/her, and at least two-thirds of the Committee members then present must vote in favour of the expulsion for it to take effect. No ex-member of the Group whose membership was termininated under this rule, or who owes money to the Group may attend any meeting,or be introduced by any other member as a visitor or guest.

7. The Committee shall have power to alter the rules but no such alteration shall take effect until it has been confirmed at the A.G.M., or a special meeting convened for the purpose.

Cont.....

8. The A.G.M of the Group shall be held during the month of May, to transact the following business:-

a) To receive, and if approved, sanction any duly made alteration of the rules.
b) To appoint Officers.
c) To appoint an Honorary Auditor or Auditors.
d) To deal with any matter which the Committee wish to bring before the members, and to receive suggestions from the members for consideration by the Committee.
e) To receive and approve the annual financial statement.

Notice convening the A.G.M. shall be sent to the members not less than 14 days before the meeting, detailing the matters to be dealt with.

9. Extraordinary General Meetings may be convened at any time by the Committee,and shall be convened within 28 days from receipt of a requisition in writing signed by 20 members or one-third of the paid up membership(whichever is the less),specifying the object of the meeting for any of the following purposes:-

a) To consider, and if approved sanction, any duly made alteration of the rules.
b) To deal with any special matter which the Committee wish to place before members, including the expulsion of a member.
c) To receive the resignation of the Committee, or to remove any member from office; to fill any vacancy caused thereby.
d) To deal with any matter which the members requiring the meeting may wish to place before the Group.

10. Notice convening an Extraordinary General Meeting will be sent to members not less than 21 days before the meeting, specifying the matters to be dealt with.

11. At all Committee and General meetings, the Chairman (or Committee nominee) shall preside, and shall have a casting vote in the event of a tied vote. Only paid-up members may vote at any meeting.

12. At all Committee meetings one third of the Committee members shall form a Quorum, of which at least two must be Officers.

13. A bank account shall be opened in the name of the TIUG(UK). The names of the persons able to sign cheques shall be authorised by a resolution of the Committee, and a mandate be given accordingly. A copy of any such resolution signed by the person holding the office of Chairman at the time of the resolution shall be supplied to the bank. Committee members will when required, be issued with Pay-In books.

continued.......

cont..........

14. If the Group is discontinued the Committee then serving shall pass the balance in the bank, after the payment of all expenses and creditors, to either an organisation deemed to form a suitable successor to this Group, or to a charity or charities nominated .

15. Nominations for election of Officers must reach the Secretary before the A.G.M. Each must be nominated by at least two members. The nominee must have agreed to accept nomination.

16. The Committee shall have the power to appoint Honorary Members, if it deems this to be in the best interests of the Group.

17. The Committee shall participate in the production of a quarterly newsletter/magazine known as "TI*MES", and the Group will circulate the newsletter/magazine to all members who have paid a subscription. Committee members responsible for dealing with other organisations shall if it is deemed to be necessary to supply these with a copy of "TI*MES", be supplied with such copies.

18. Committee members responsible for dealing with Group facilities (i.e.Libraries etc) shall so conduct matters that that service shall be a self-financing entity. Charges must be agreed by the Committee Officers. Any monies held as an agreed deposit against materials loaned to members shall be paid into the Group bank account, and will be retained by the Treasurer until notified by the Committee member concerned that either:-

a.The transaction(s) are now satisfactorily completed, and the deposit should now be returned to the member.

b.That the deposit should be reduced by a sum determined by rule as a result of the members failure to comply with the agreed facility rules of conduct.

or

c.That the material on loan has not been returned in a useable condition, and the deposit should therefore be forfeit to the owner of the material concerned.

The Committee member paying in such deposits should inform the Treasurer within a reasonable period of the name of the member making ther deposit and the sum involved so as to enable the Treasurer to maintain his control over financial matters.

19. Officers shall be empowered to make decisions becoming necessary for administrative reasons, such decisions must be agreed by three members, and will be open to challenge by the Committee members. This provision will include the setting and approval of budgets.

20. Any Committee member, or any Group member via a Committee member may require the Secretary to place before the Committee a POSTAL PROPOSITION, The following procedure will apply:-

a. The Secretary shall put the proposition to a panel consisting of the Chairman, Vice-Chairman, Treasurer, Secretary and the Committee Member responsible for dealing with the subject under consideration.

b. Having considered the proposal, panel members should then make their decision known to the Secretary, who must then notify all Committee Members of the result, including the details of voting in the event of a split decision.

c. The decision taken by the panel will be effective subject to approval, rejection or modification at the next meeting of the Committee.

Your Letters

Dear Editor,

Obituary.

I feel I would like to inform you of the death of one of your subscribers, Mr. R.W. Johnson, 8 Ullswater, Bracknell, Berkshire, who was aged 62. I find my brother has been a subscriber to your magazine for quite a long ti'me. He was an enthusiastic user of his Texas equipment, also a very clever mathematician.

If you could perhaps place an obituary in your next issue it would notify any person I have overlooked among his correspondence. I am trying to write personally to.those he was. corresponding with,

Yours truly,

Mrs. J.A. Atkinson (sister to Ron Johnson),
9 Harcourt Way, Abingdon, Oxfordshire OX14 1NV

****

Dear Christina,

I am writing to you so you can pass this on to other members who are not lucky enough to have a Disk Drive (I only have Extended Basic, 32K expansion, and Mini Memory. I wrote to Maurice Rymill to ask for machine code programs on cassette because all good programs are on disk, which I haven't got. Maurice put me in touch with Stephen Shaw who told me that he could load disk pronrams on to tape. He sent me a list of programs that might work on tape using Ext Basic plus 32K expansion. I wrote hack to Stephen, and in a couple of days he sent me a LOADER-LOAD program which you need to load machine code programs, plus two games programs.

I loaded the LOADER-LOAD program into the computer, then loaded Star Trek. This had to be done in three parts, but was no problem. The game started straight away, no waiting as with other Basic programs. The graphics and speed of control were great! The other program, Defender, loaded the same and worked first time. Star Trek is an 18K program, Defender is slightly smaller. There are over 50 programs on the list Stephen sent me, all in machine code. If you have Editor/Assembler module you can load even faster without using the LOADER-LOAD program. The price of £1.50 for three programs is rather good when the same games on a module are about £10-£15 each,
Mr. K.F. Hughes, 220 Broadland Drive, Lawrence Weston, Bristol, Avon, PS11 0PN.

SUPER SPACE II MODULE - A REVIEW

by E.H.Shaw

The Super Space module from DataBiotics is a powerful tool for programmers and non-programmers alike. It consists of an Editor Assembler grom (6k) plus four 8k banks of battery backed ram (32k). Only one of the four banks can be enabled at any one time. Programs in the module are run by selecting them from the power-up screen:

PRESS	FOR
1	TI BASIC
2	EDITOR ASSEMBLER
3	YOUR PROGRAM NAME

You are allowed up to seven user defined names of up to twenty characters each. This is done by preceding programs with a ROM header and solves any problems of linking via Ref/Def tables etc.

The Package

The package as ordered from Tex-Comp consisted of the following:-

Super Space Module
3 Flippy Discs
1 Manual
2 Books (Intro to Assem Language for the TI Home Computer - Molesworth
Programs for the TI Home Computer - Steve Davis)

The Software

One flippy disc is devoted to the Super Space Utilities, the other two contain Art Green's Macro Assembler and Clint Pulley's C99 which I shall not discuss.

The main utilities provided are:

A version of John Johnson's Menu program to run out of Super Space.
A similar Menu program by Clint Pulley which would load editor and formatter directly.
The source code for item 2.
A basic prgram to write your own menu program to load any type of assembly code, not just programs.
Three sample ROM headers.
A bank loader program to load four program files one into each bank and then execute the first one.
A program to save the contents of any ROM module to disc so that they can be reloaded and executed in Super Space.
A stand alone file of editor assembler utilities.
Lines and circles demonstration program.

Phew!!

If you use the menu programs provided then these run straight away in the module and instructions for configuring to your set-up are clearly given. If you write your own programs, all you have to do is to make sure that the header (provided) is included with the source and that the code starts with AORG >6000. The programs are assembled as normal and loaded using the editor assembler part of super space. It does not matter whether the code is DF80 or program format. The name of the program will appear on the power up menu screen and is selected from there. If you use the bank switching facility then four programs are assembled as above, all starting at 6000 hex but this time also containing some code to switch between banks. The four are save to disc and loaded in one operation using the bank loader utility provided.

The only area where I ran into trouble was in saving modules to disc. I could not get this to work at all, but then I am not sure which modules contain ROM only so perhaps I was using the wrong ones.

I found that super space was entierely compatable with the Miami Users Group Operating system v6.3 for the Horizon Ram disk and also with the version of the menu program provided with this. The 's' key toggles between Editor/assem and your program in ram.

I also found no problems in running Funnelweb v4.0 from the cartridge using the file CT8RAM which is provided for this purpose.

Ease of Use

I feel that to write ones own programs and especially to deal with bank switching requires in depth understanding of assembly language. You could write a complete operating system for the 99/4a bank switched to reside in the 32k module. However, even if you only use the programs provided or Funnelweb the module still has impressive capabilities.

The Verdict

The total cost to import this module into the UK in January was £49.73. This included £10.34 customs duties. The package is quite a heavyweight when all the software which is supplied is taken into account and in my opinion represents extremely good value for money.

TIMES

RAMBLES
Welcome to the July 1989 issue of Rambles by Stephen Shaw.
Your letters are always welcome, on any subject. Ask me anything you like, I can always say I know nothing, but you won't know if you don't ask! Anything you'd like to see in Rambles perhaps? The address follows, and is the same address as your group's disk based library — send two disks and return postage for a copy of our disk library list in D/V 80 format.

10 Alstone Road, Stockport, Cheshire SK4 5AH, United Kingdom.

CHANGES, CHANGES ...
If you look back to Issue 19, you, will see a notice inside the front cover that we are not affiliated to TI, etc. That notice was removed from Issue 20.
This does not mean we are now affiliated to TI, etc.
This is a history lesson ... when the first UK TI user group was founded, Texas Insruments Inc. were extremely protective not only regarding information on their new computer, but also on the use of their name, including the use of the initials TI within the magazine name, which was then TIdings. Their legal department insisted on the disclaimer and who are we to argue with a corporation that big! We are not suggesting that ANY user group was EVER ever affiliated to TI — we were merely doing what TI insisted we do.
Since those balmy days, TI has every appearance of having lost interest in their home computer, its programs, and even their initials! And, as one UK user took umbrage at our using the enforced disclaimer, we have dropped it to avoid causing offence. (Yes, I know every good UK citizen regards the initials TI to stand for Tube Investments, but Texas Instruments was very, very protective back then!)
You can spot the older user groups by looking for a non-affiliation disclaimer in their newsletters.

Issue 20 was very well printed! This time I am printing Rambles again in Elite font, but this time just single strike. To improve the density of the print I am — for the first time — using a carbon film ribbon. Yes, a carbon film ribbon for the Epson FX80 / MX80 series of printers! Naturally it is not so economic as a fabric ribbon, but the dots are all very black! And again, I shall not be pasting the work up, so any blank spaces, contact the editor!
(2022- issue 21 issue was so pale I have had to retype this section.... sigh)
==

CORRECT TO N PLACES....
Rounding is easy. Just use:
```
    RESULT=INT(NUMBER+0.5)
```
But what if you want to display a number correct to a certain number of digits?
A little more difficult but not impossible. The GENERAL format is:
```
    RESULT=INT(NUMBER * P +0.5)/P
```
Where P=the power of 10 of the number of places to be rounded.
That's easy isn't it? To round to two places: P=100   (10^2) !
The general format will only work for positive numbers. For numbers which may be either negative or positive, the formula. becomes:
```
    RESULT=INT(ABS(NUMBER)*P+0.5)*SGN(NUMBER)
```
Here is a tiny utility subprogram for you for TI Extended Basic.
Just remove the first and last lines and GOSUB instead.

```
100  SUB PLACES(NUMBER,PLACES,RESULT)
110  P=EXP(PLACES*LOG(10))
120  RESULT=INT(ABS(NUMBER)*P+0.5)/P*SGN(NUMBER)
130  SUBEND
```

Now to find out what 23456 rounded to 3 places is you would use:

```
1 CALL PLACES(23456,3,RESULT)
2 PRINT "Result" :RESULT
3 STOP
```

– Maurice E T Swinnen of Mid-Atlantic 99ers. 1986

==

MACHINE CODE FROM TAPE:
Did you know that CALL INIT:: CALL LOAD("CS1") works! Of course first you have to transfer those DF80 files from disk to cassette. And some of the lines may need shortening (you do remember that cassette data files do not use 80-byte records!), dropping off the checksum is a good start there!

```
e.g. from       ...9BD3A87F131F     0001
                ...5A47F196F        0002
                ...A127F1BFF        0003
shorten to:
                ...9BD3A8F          0001
                ...5A4F             0002
                ...A12F             0003
```

i.e. remove the 7 tag and the next four characters. Leave the final F.

Read each record from disk and transfer the first 64 characters to tape. You will need to note how many lines are to be transferred or use the EOF function. The tape file is saved as DISPLAY/FIXED 64. You will also need to know the starting name to LINK to.

Then of course, there is the time element. Take a look at how many lines your average D/F 80 file has. A 54-line file, loaded from tape, takes 6 minutes. Now work out how long it is going to take to transfer Super Demon Attack this way, and where are you going to find a cassette that long!

Just one more capability the TI-99/4A has, that has not been documented.

Now we have the Funnelweb LOAD program, tape owners can find it much easier to load machine code into their 32K RAMs using memory image files. Much faster.

Editor/Assembler module owners can just use the LOAD PROGRAM FILE option and indicate CS1. It really was clever of TI; the ONLY loader that TI supplied to enable cassette owners to load machine code into expansion RAM and it is sold on a high-priced package; module, huge manual, and two disks.
– Thanks to George Meldrum, TIsHUG, May 1988.

===

SOURCE CODE. Author and original publication not known.

Assemble this little utility into a non-compressed D/F 80 file and you can load it into XB with a CALL LOAD. Then it will auto-boot "DSK1.LOAD" whenever a running XB program breaks for any reason – a little more powerful than ON BREAK NEXT. Try it!

```
          DEF     CHECK
CHECK     MOVB    @>8344,@>8344     * >8344 IS 0 IF XB NOT IN RUN MODE
          JEQ     NORUN
          B       *R11              *  EVERYTHING OK SO RETURN ELSE:
NORUN     CLR     @>83C4            *  TURN THIS ROUTINE OFF (ISR HOOK)
          LI      R1,>6372          *  XB GROM START
          MOVB    R1,@>5C02         *  MAY NOT WORK WITH SOME XB VERSIONS
          SWPB    R1                *  WRITES >63 TO GROM WRITE ADDRESS REGISTER
          MOVB    R1,@>9CO2         *  WRITE >72 TO GROM WRITE ADDRESS REGISTER
          B       @>006A            *  EXECUTE XB
          AORG    >83C4             *  PLACE THIS ROUTINE'S ADDRESS AT >83C4
          DATA    CHECK
          END
```

==

"We do not know what is going on in England. we know the Queen is still kicking and alive (God save Her!), but we are afraid the TI-99/4A is dead there." Huh? Written by Paolo Bagnaresi, Milan, Italy, in August 1986, to Ottawa, Canada. Guess our reputation hasn't got to Italy just yet then!
==

Infocom adventures are not all fully logical – there is a random element in some of them which means that sometimes you die and sometimes you live.
An interesting command to type in to your Infocom adventures is $VE. Try it!
There is a #RAND command in Lurking Horror, which expects a number before/after it, not sure what it does, but I think it may determine the path when you come to a random choice. Hitchhiker has a total vocabulary of 969 words.
Have you found them all yet? KILL ADAMS? Some odd commands, purpose unknown include XYXXY and ZZMGCK – the latter may just be an end of file dummy.
Suspended has a vocabulary of 680 words, but you can complete it with just 35, that's real overkill!
==

MICROpendium
If you do not yet subscribe to MICROpendium, why not? The cost is reasonable.
Prices advertised in my March 1988 copy (I get mine by sea mail!) are USD $23.50 per year by sea mail and USD $37.00 by airmail.
Send USD International Money Order, bought from Lloyds or Barclays Banks, to:
MICROpendium, PO Box 1343, Round Rock, TX 78680, USA.
This is no fly-by-night magazine. The 1988 March issue is their 50th monthly issue.
Regular articles on Basic, c99, Geneve, and plenty of reviews. Back issues are $2 each by sea mail to subscribers only.
==

D/V 80 FILES THAT TI-Writer CAN'T read
Any time you see a DISPLAY/VARIABLE 80 file on a disk, it is always a good idea to take a look at it with TI-Writer, as there is a good chance it is either documentation or source code – which may contain documentation. In almost every case a D/V 80 file should load with TI-Writer, whatever is in it, text or data or anything.
However, there are a few text files coming out of Europe that you CANNOT load with our version of TI-Writer – our European friends are using a different and incompatible version of TIWriter – v 2.0. If their text files are saved to disk with PF there is no problem, but using SF adds tab information.
In order to produce those odd European characters, printers use ASCII codes outside the usual range of v1 of TI-Writer, so v2 was modified to accept them, and the tab info had to be modified as well. And as their tab data is outside the capabilities of our version to handle, the result is a console lock-up.
Curing the problem was difficult, until our membership secretary dealt with it.
In the 1988 April issue of EAR 99ers' newsletter, he presented a program to amend the tabs on a v1 TI-Writer file – largely to demonstrate the way the tabs are saved. What was interesting was that by appending a new tab set, the original set js "replaced". If it works for v1, why not try appending a v1 tab set on to the end of a v2 file?
It works! The following routine is a much modified form of his program.
If you can't load any D/V 80 file, amend it with this trifle and try again!

```
100 REM MAKE V2 TI-WRITER FILE ACCEPTABLE TO V1
    based on an idea by Peter Walker. UK. 1988.
110 DISPLAY AT(2,2)EPASE ALL
:"INPUT NAME OF FILE TO BE":
"MODIFIED:"
120 DISPLAY AT(6,2):"DSK1."
130 ACCEPT AT(6,5)SIZE(-12):
FILE$ :: FILE$="DSK"&,FILE$
140 OPEN #1:FILE$,D1SPLAY,VA
RIABLE 80,APPEND
```

```
150 A$=CHR$(128)&CHR$(134)&C
HR$(128)&CHR$(212)&RPT$(CHR$
(213),16)&CHR$(128)&CHR$(136
)
160 PRINT #1:A$
170 CLOSE #1 :: PRINT "DONE"
180 END
```
===================

PR BASE HINTS:
 Sorting and so on are based on an ASCII STRING and everything works according to the ASCII values of the characters. Thus while 4 comes after 2, 22 will come before 4. Use leading zeroes on numbers you are sorting by- then you will correctly find the sort as 02,04,22.
Selective Indexing search works on your input UP TO the first space, so that "good day" will only work on "good". To use the whole thing, you must insert a question mark, thus "good?day".
PLEASE will someone write tutorials for us for PRBASE and CFS!
==================

AXIOM PRINTER?
 By this time some owners of AXIOM printer may discover that their printers are not entirely behaving themselves. In particular, your computer may give you error messages indicating the printer isn't there! The problem lies in the thin fiddly wires they used which are PUSH FITTED in their little connectors. In due course the wires will either pull out, or more likely, break near the end, and contact is lost.
 The solution is to remove the whole ribbon, remove about a half inch off the end, and then refit- soldering is recommended, but CAREFULLY.
===========

MINI MEMORY PROBLEMS?
 As the battery wears out, the voltage drops, and in the end, the dead battery actually stops the module working- it will not retain data even inserted in the console, even though the battery is only for "back up". A quick solution is to open the module up and snip one of the battery wires- the module will now work in the console! For battery back up, take advantage of the offer advertised in this magazine for several years now for ni-cad replacements.
======

CASSETTE CARE:
 The Manchester Central Library have published an excellent leaflet on care of cassettes and cassette players, which is highly relevent to Cassette Users. I have extracted the juicy bits for you...
MUSTS:
1. CLEAN your recorder regularly- if you use a "wet cassette" you must still clean separately the capstan and pinchwheel, as these special cleaning cassettes only clean the heads properly. See notes later!
2. Keep cassettes in cases, away from heat and magnetic fields and damp.
3. Before putting a cassette in the machine, rotate a spool with your finger to make the tape fully taut.
4. Never use C120's and for best results stick to C60s or shorter. C120s are very likely to stretch and snap, and even C90s may come apart quite quickly. The thinner tapes are more likely to snarl up your machines.
5. Cassettes are not hammers. Observe care.

Detail:
A cassette tape is a VERY thin thing, coated with aneven thinner oxide coating, and even under the best conditions, the best tape will shed its coating little by little... onto the surfaces of your recorder, where they stick and begin to scratch your tapes and even more coating comes off....

The PINCHWHEEL is the larger rubber-like wheel which rotates pulling the tape onto the take up spool, while the CAPSTAN is the thin metal wheel which presses the tape against the pinchwheel. If you dont clean these properly, tape is liable to snarl up in your player. The only way to clean them is with COTTON BUDS, dipped in special head cleaning fluid or meths. Some alcohols may be suitaable but NOT isopropyl alcohol, widely used for cleaning disk drives!, as isopropyl alcohol can cause uneven swelling in the pinchwheels.

Clean the heads to avoid undue wear, and losses of audio quality (dullness) which may stop the player being computer compatible.

CLEAN EVERY 10 HOURS PLAYING TIME.

If you use a head cleaning tape, discard it after ten uses, after that it just spreads dirt around.

Tape heads become magnetised over a period of time- as the tape passes over them, the heads gradually build up a magnetic charge from the tape. This could produce increased background hiss, and might eventually result in tapes becoming partly (and irrevocably!) erased while being played. Demagnetise your heads AT LEAST once a year. Special cassette-shaped devices can be bought and are safest to use.

(With thanks to Manchester City Council-Cultural Services)

==========

ATARISOFT BUG:

In case you just find one, the early Atarisoft modules for the TI99/4A do not all work properly in SOME consoles. Atari downloaded the "large character set" from grom 0, but used a direct address instead of the indexed address, and yep, some consoles are 4 bits out on the direct address. It means those characters look kinda Japanese...

=====

MILTON BRADLEY BUG (and others):

Some MB games intended to use speech have a bit of a problem when loaded from disk dumps into the 32k ram. The delay in testing for speech is just not long enough (see EdAs manual pages 349/350), so sometimes the games will fail to identify that you have a speech synth, and give no speech.

======

TI FORTH DEBUG

The following corrections have just surfaced from Ottawa- passed on without testing. Change a COPY of your Master Disk!

```
  Screen 58 line 10 to read:
  VDPMDE @ 4 < IF SMTN 80 0 VFILL 300 ' SATR ! ENDIF
  Screens 53,54,54 line 1 should end SETVDP2.
  Screen 59 line 9 change 00FF to 00FE
```

============

TENEX are beginning to upset me. I wrote in Feb 88 asking for a catalogue. No reply. I wrote again in April for a catalogue. No reply. Last chance: wrote again, May 16th 1988. Result: catalogue POSTED May 17th received May 26th. WITH NO ENVELOPE OR WRAPPER and nothing holding the outer edges of the pages together. Delivery address and stamps at the bottom of the back cover. Delivery can best be described as a miracle- fortunately they wrap products up a lot better. If you write, I suggest you ask as strongly as you can that your catalogue is honored with an ENVELOPE wrapped around it!!!

Again some nice goodies, and order sent airmail May 31st, including a couple of copies of SPAD XIII for members, who asked for them at the AGM.

Have sent for a freeform database, an alternative spreadsheet, and one or two other goodies, watch for reviews as they arrive here in Stockport.

Prices of many products continue to fall. TENEX have a good range of NEW items, both disk and module software and peripherals.

For older modules, TEX COMP in California LIST the most, while PARCO has good stocks of the older modules and even some of the very latest.

==========

WHAT'S IN A FILE?
Disks for the TI can have a number of different file types, which can serve several different purposes. How can you tell which is which? In some cases, you are reduced to trial and error! But the following notes may help:

PROGRAM FORMAT:
Unwisely named, as not all PROGRAM files are programs. This type of file is better described as MEMORY IMAGE- just a byte by byte image of a particular area of memory in the computer, which MAY be a program in basic or machine code, or some kind of data- graphics, adventure data and so on.
FUNLWEB will identify basic and most machine code images for you- use SD from TI Writer, and once the directory is on screen press the equals (=) key. The right hand column will now be marked BA or EA as appropriate.
Other memory image files can be identified as follows:
File name ends in _P or _C: A picture to load with TI Artist or MAX/RLE.
File is 54 sectors long: MAY be a picture to load with Graphx or MAX/RLE.
File is 25 sectors but doesn't end in _P or _C: MAY be graphics for CERT99.
File is 8 sectors: May be a graphics screen for Fractal Explorer.
File names are identical except for last letter:

```
  Type 1:                          TYPE 2:
  INVADERS                       INVADERS
  INVADERT                       INVADERS1
  INVADERU                       INVADERS2
   =Ed/As Option 5                 =Gram Kracker
    Machine Code program            Machine Code program.
```

If you place the files on a blank new disk, and inspect the header (first 3 bytes) on Sector 22, with a sector editor, as supplied with Funlweb, you may also determine:

```
  TUNNELS OF DOOM:  0406 0504 0400
  Scott Adams Adventures: 2020 2020 2020
  Module ram required: If third byte is >8000 or >7000
```

If the first byte is 0000 you MAY be dealing with graphics such as a CHARA1 file or a GRAPHX picture.

On standard E/A Option 5 files, where there is more than one file, eg INV1,INV2 and so on, the first byte on the LAST file is >0000 while all preceding files have a first byte of >FFFE (Gram Kracker files are >FF05,>FF06 etc).
***
DISPLAY VARIABLE 80
Almost all DV80 files can be loaded into TI Writer for inspection. If they won't load they may have been prepared with the European version of TI Writer, just use the little utility program given elsewhere in this issue to force them to load!
DV80 files are most commonly:
TEXT- documentation.
SOURCE CODE - which may contain instructions!
GRAPHICS:
Files ending _I,_F,_S are for TI ARTIST.
5 sector and 2 sector files MAY be for Picasso.
Files may be RLE graphics.
***
DISPLAY FIXED 80:
Usually used for machine code object files, to load with XB or EA modules. Can also be loaded into TI Writer for inspection.
36 and 68 sector files MAY be fonts for THE PRINTERS APPRENTICE.
***
DISPLAY FIXED 128:
Used for RLE GRAPHICS.
MAY be ARCHIVED files which need unpacking.
Could be a special format machine code file requiring a special loader.
***
INTERNAL FIXED 128:
Used for JOYPAINT FONTS.
Also used for COMPRESSED ARCHIVED files which need uncompressing.
***

DISPLAY VARIABLE 163:
 Used for Extended Basic MERGE files.
 type MERGE DSK1.FILENAME.
***
DISPLAY FIXED 254 is used for Draw a Bit graphics.
***
INTERNAL VARIABLE 254 is used for:
  LONG Extended Basic programs.(Use OLD DSK1.FILENAME as usual)
  Data for Creative Filing System
  Data for TRIO SINGS program.
  Data for CSGD- watch for file names ending in /CH and /GR etc.
***
DISPLAY FIXED 255:
Used for INFOCOM data files, usually GAME1 and GAME2.
Used for Super Disk Cataloguer data files.

That will give you a start anyway!
==

CORRECTION: ISSUE 20 : PAGE 42.
 Beamheadings Program: Hmmm. Can't find any 1/2 or 1/4 keys on my keyboard... these odd characters, inserted in the listing by a daisywheel printer, should be characters 123 and 124 respectively- you noticed them redefined at the start of the listing didn't you! Sorry 'bout that- program was printed by us in the form it was received!
=====
SECOND HAND PRICES
What is a good price for second hand geear? What somebody will pay for it... seriously, I have seen adds quoting prices which seem intended to stop anyone even haggling, so high are the prices asked.
 If you are selling TI gear, consider what YOU would NOW pay for it yourself! When five hundred will pay for a complete computer with monitor disk drive and eerything else, with high res graphics, faster processor and so on than the TI, why should you expect a TI owner to lay out nearly a thousand on a fully fitted PEB! Everything you are selling is likely to be AT LEAST five years old and probably heavily used! May I suggest a limit of 25% of purchase price for second hand goods- many will be worth very much less than this though. New games and educational modules are readily available NEW for under three pounds!
 AND if any members ARE giving up their TIs (shame shame) please do consider donating your equipment to the Group, in support of remaining members.
=====
FREE DRAW : DISK OWNERS ONLY
Why on earth do I tell you about new products....
Long long ago I mentioned a new "diskazine" called the Genial Traveller, issued "bi monthly" by Barry Traver. For various reasons, including ill health, the first volume took a little longer than a year to get through, but already volume two is in full swing, at the silly price of US$35 for six flippies. Barry does not give a separate overseas price, but $40 would no doubt be welcome.
... Barry has written to tell me that I was the ONLY subscriber in the UK. To encourage a little interest he has sent over a complete set of the first volume (nine flippies - not bad for a six issue sub huh?). And the complete set is available in a draw which is FREE but ONLY available to members of THIS group.
 TO ENTER the free draw, send two disks with return p&p for a copy of the library catalogue. Include on one disk a DV80 file containing your name and address (important! will be used to select winner!).

 The draw will take place August 30th, and the winning name will be checked with our membership secretary for fully paid up status! The lowest possible value is 9 flippies- but these are pretty full of quite a variety of material.
 NOTE: Anyone receiving Traveller in a pirated format is requested to consider a PAID FOR subscription. It is not issued as freeware but as a commercial issue, even though freeware and public domain programs may be included. The price is so low that piracy is not only hurtful to Barry but daft in the bargain!
Barry Traver, 835 Green Valley Drive, Philadelphia, PA, USA, 19128
=============

News of further cable problems, this time the hefty cable to the PE Box, which like the Axiom cable is liable to fractured wires if mistreated or subjected to frequent movement. And if your console moves around a lot...
 A possible solution... BOOTS have opened a section for the disabled (ALL branches can order all products if not stocked) and one interesting item is a non-slip mat made by Dycem. It seems to be some sort of very grippy plastic, and has its main use as a bottle grip for opening bottles, but also useful for trays and so on. And very suitable to place between a console and a table top, to very effectively stop console movement. Essential if small keyboard users are around, and essential for stand-alone users. Try it!
 The Dycem product is not to be confused with so-called non-slip rubber mats you can get for typewriters, which very often are more slippy than nothing!
It is VERY grippy!

Macroassembler?
 The disk library continues to hold the Macro Assembler by Art Green, but what is a macro?
 Example of use. Your source code frequently has code like:

```
      CI  R0,168
      JL  LABEL1
      B   @LABEL2
LABEL1 ....
```

- that is, check a register value, and jump depending on its value.
   Using the macro assembler, you can use a supplied macro, IF, as follows:

```
      IF  R0,GE,168,LABEL2
LABEL1 ...
```

   Why? It can ease the writing process! This assembler is faster in operation than TI's, and also has a nicer LISTing format.
 (sample code from Mike Dodd, K Town 99ers)
============================
 As there seems to be a lot of reinventing of wheels going on in the States, with a few "exciting" new offerings not being quite as good as what has gone before, may I draw your attention to programs in the Disk Library which can:
  LOAD a DV80 file into a running program FAST.
  RUN a program LISTed to a DV80 file!
  Change compressed/uncompressed DF80 object files to the alternative format.
  Remove the autostart on some DF80 files.
  Embed a DF80 file into an ordinary XB program for cassette loading!
  Move a PROGRAM format machine code file onto tape ready to be loaded with EdAs module or, using Funlweb, using XB.
  Create a PROGRAM format m/c file from a DF80 file, without having to insert SFIRST, SLAST, and SLOAD labels.
  Extract lines from an XB program FAST.
  Set an alarm clock to remind you when to finish computing...
  and so much more....
  Take another look at the disk library listing, and advise your needs!
==================
  TI made 250 TI99/8's. How many TI99/5's did they make? (The answer is NOT none!). A complete set of "home computers" from TI would be quite a collection... especially if you collected all the different versions of the TI99/4A as well...
====================
 PROGRAM SEARCH:
 Here is a tiny program which will search through an XB program for a variable name, keyword, or quoted string, or line number, or just about ANY text- for example, FOR I=1 TO or CALL KEY(3 or IF A>B AND..... up to 100 occurences will be listed for you.

The list on the left is in XB and should be saved in MERGE format.
To find a seçtion of an XB program, load the program, then MERGE this in to it.
Then type:
32000 and the thing you are searching for, eg
32000 "A" (string only)
32000 A   (string + var. name)
32000 GOSUB (all GOSUBs)
there are a few things XB will not permit you to type in. There is also an exception- line numbers.
To find a NUMBER 100 use:
32000 100
to find any reference to LINE NUMBER 100 however (for instance, GOTO,GOSUB, IF, etc) you must use:
32000 GOTO 100
The first entry of GOTO on line 32000 is ignored. To search for all GOTOs, use:
32000 GOTO GOTO
...
The program:

```
1 CALL FIND :: !@P-
32000 !
32010 !@P+
32020 SUB FIND
32030 ! SEARCH FOR CONTENT
OF LINE 32000
32040 DIM A(100)
32050 ON ERROR 32080 :: CALL
 LINK("FIND",A()) :: ON ERRO
R STOP
32060 FOR I=1 TO 100 :: IF A
(I)=0 THEN STOP
32070 PRINT A(I); :: NEXT I
:: STOP
32080 CALL ERR(B,C) :: IF B=
84 THEN PRINT "ERROR: INSERT
 PATTERN IN 32000" :: STOP
32090 IF B=57 THEN RETURN 32
060 ! SUBSCRIPT ERROR
32100 IF B=14 OR B=135 THEN
CALL INIT :: CALL LOAD("DSK1
.FIND/OB") :: RETURN 32050
32110 RETURN
32120 SUBEND
========
```

This is the source code which is to be assembled into the file FIND/OB:

```
        DEF  FIND
START   EQU  >8330  * START OF LINE NO TABLE
FINISH  EQU  >8332  * END OF LINE NO TABLE
XMLLNK  EQU  >2018  * FOR EX BAS USE !!!
NUMASG  EQU  >2008
FAC     EQU  >834A
GOTO    BYTE 134,0  * Filter GOTO
FIND    BLWP @FIND1
        RT
FIND1   DATA FINDWS,FIND2
FINDWS  BSS  32
FIND2   MOV  @START,R1
GETPAT  BL   @MOVE
        CI   R6,32000 * PATTERN IN LINE 32000
        JEQ  GOTPAT
        JL   ERROR
        AI   R1,4
        JMP  GETPAT
ERROR   LI   R0,>2100   * DATA ERROR
        BLWP @>2034
GOTPAT  INCT R1
        BL   @MOVE
        MOV  R1,R8
        MOV  R6,R9
        DEC  R9
        CLR  R10
        MOVB *R9+,R10 * PATTERN LENGTH TO R10
        SWPB R10
        CB   *R9,@GOTO * GOTO FILTER
        JNE  NO
        INC  R9
        DEC  R10
NO      MOV  @FINISH,R2
        DEC  R2
        CLR  R0
LOOP    C    R2,R8
        JEQ  DONE
        MOV  R2,R1
        BL   @MOVE
        MOV  R6,R3
        DEC  R3
        CLR  R15
        MOVB *R3+,R15
        SWPB R15
LOOP2   DEC  R15
        JEQ  NOMTCH
        MOV  R3,R4
        MOV  R9,R5
        MOV  R10,R12
```

It IS possible to transfer the machine code into CALL LOAD format, and add it to the MERGE file, but on a long XB program, this adds considerably to both the time taken to MERGE the file in, and then to RUN it.

This code is from
STEVEN KARASEK and was first published in:
COMPUTER BRIDGE
Vol 5 No 11 November 86

=========

The utility uses the on-board tokenisation facilities on line 32000 and compares the tokenised form to the entire program in memory.
NOTE: If your basic program contains a line with !@P+ in it, those symbols will almost certainly have to be removed in order for this utility to run properly!

```
LOOP3  DEC  R12
       JEQ  MATCH
       CB   *R5+,*R4+
       JEQ  LOOP3
       INC  R3
       JMP  LOOP2
MATCH  MOV  R2,R1
       DECT R1
       BL   @MOVE      * LINE NO TO R6
       MOV  R6,@FAC    * TO F/PT ACCUM
       BLWP @XMLLNK    * TO F/POINT NUMBER
       DATA >20
       INC  R0
       LI   R1,1
       BLWP @NUMASG    * LINE NO TO BASIC ARRAY
NOMTCH AI   R2,-4
       JMP  LOOP
DONE   RTWP
MOVE   INC  R1
       MOVB *R1,R6
       SWPB R6
       DEC  R1
       MOVB *R1,R6
       RT
       END
```

Retyped by sjs. Queries/complaints to S Shaw!

Repeated use of this program alternating with running of your XB program is possible just by removing lines 1 and 32000 to RUN YOUR XB program, and by re-inserting lines 1 and 32000 to again use the utility. The CALL LOAD will not be repeated provided your program does not have a CALL INIT in it... this will save on the frustrations of the speed of MERGEing....
++++++++++++++++=========+++++++

WHATS NEW
PUSSYCAT

Look ... and
find out for
yourself!

Here are some interesting routines by Dr Roy T Tamashiro, for XB plus 32k.

```
FLIP.
Lines 100 to 150 are a DEMO
program. The FLIP routine is
in lines 30100 onwards and
may be saved as a merge file

100 CALL FLIP
110 CALL CLEAR
120 PRINT "FLIP DEMO": : :
130 FOR I=33 TO 126 :: PRINT
 CHR$(I); :: NEXT I :: PRINT
140 CALL LINK("FLIP")
150 INPUT "Press enter":R$ :
: GOTO 110
160 !
170 !
30100 SUB FLIP
30110 ! 1986 R TAMASHIRO
30120 ! COMPUTER BRIDGE
30130 ! NOVEMBER 1986
30140 ! XB + 32K
30150 !
30160 CALL INIT :: CALL LOAD
(16376,70,76,73,80,32,32,36,
244) :: CALL LOAD(8196,63,24
8)
30170 CALL LOAD(9460,2,0,4,0
,2,5,37,72,2,2,0,2,2,4,0,4,2
,1,40,56,4,32,32,44)
30180 CALL LOAD(9484,192,96,
40,56,6,5,213,65,6,193,6,5,2
13,65,5,192,2,128,6,240,21,5
,6,4)
30190 CALL LOAD(9508,22,239,
2,37,0,16,16,234,2,2,2,240,2
,1,37,64,2,0,4,0,4,32,32,36,
4,96,0,112)
30200 PRINT "FLIP LOADED"
30210 SUBEND
30220 END
```

```
UPSIDE DOWN DEMO
(-Merge FLIP into it!!!-)
100 CALL FLIP ! Merge it in!
110 DATA 1,"UPSIDE DOWN"
120 DATA 3,"by Roy Tamashiro
"
130 DATA 5,"from Computer Br
idge"
140 DATA 6,"November 1986"
150 DATA 15,"After a while,
everything"
160 DATA 16,"gets corrected.
.."
170 DATA 18,"It will wait fo
r you to"
180 DATA 19,"press the ENTER
 key"
190 DATA 99,"DUMMY DATA"
200 RESTORE :: CALL UPSIDE
30000 GOTO 200
31500 SUB UPSIDE
31510 DIM W$(24) :: FOR I=1
TO 24 :: W$(I)="" :: NEXT I
31520 READ R :: IF R<25 THEN
 READ W$(R) :: GOTO 31520
31530 CALL LINK("FLIP") :: C
ALL CLEAR :: FOR I=1 TO 24
31540 DISPLAY AT(24-I,14-LEN
(W$(I)/2):W$(I)
31550 NEXT I :: CALL DELAY
311560 FOR I=1 TO 24 :: CALL
 HCHAR(I,1,32,32) :: DISPLAY
 AT(I,14-LEN(W$(I)/2):W$(I)
:: NEXT I :: CALL DELAY :: C
ALL LINK("FLIP")
31570 DISPLAY AT(24,1):"PRES
S ENTER" :: CALL KEY(0,K,S):
L IF K<>13 THEN 31570
31580 CALL CLEAR :: SUBEND
31590 SUB DELAY :: FOR D=1 T
O 1000 :: NEXT D :: SUBEND
31600 END
```

===

The ANNUAL MEETING went off well (I quite like Derby! this year there was even a train DIRECT from STOCKPORT to DERBY! No such luck getting back, fortunately one of the Committee gave me a lift back - even more fortunately as I later found that trains from Sheffield to Manchester were NOT arriving at their advertised destination on that day due to track relaying! (Thanks).

The turnout was probably pretty good considering our membership - perhaps we can look for recruiting a few more members to support us next year!. It was nice to see Francis Parrish once again, and have a good chin-wag. PARCO are still trading, and still have good stocks of the older modules, and limited stocks of the newer ones.

Peripherals and expansion systems are clearly in demand (even if only limited demand, demand exceeds supply). Non-attendees missed some first class bargains, and some ridiculous offers were not taken up (does everyone really have everything there is now? Or is 10p just too much?).

Many thanks to Parco for subsidising the costs of the event.

Nicer still to see the Geneve working- pity the highest-res graphics disks turned out to have been wiped (aaaagh!) but the display on the Phillips monitor was still very impressive. The 80 column word processor and Multiplan were also very nice. Pity about the cost...

Good also to see Peter Brooks, a gentleman who has supported the TI in this country for about as long as anyone possibly could. And in addition to the committee members, managed to meet one or two members also, but as usual, I probably didnt speak to as many as I should have done, and made a botch of talking to those I did - I don't shine at this sort of thing, sorry!
A very pleasant day. We must get together again sometime.....
===========
If anyone wants the KJ Bible as text on disk, contact:
Raymond K Hamilton, Rt #2, WILDER, Idaho, USA, 83676
- around 70 SSSD disks! Dont forget to send $$$ or IMOs when writing.
=======
Q: How do I transfer programs from CASSETTE to DISK?

a. Before loading the tape, type in:

```
CALL FILES(1)
NEW           then
OLD CS1
```

b. If it loads but wont run, you need to use XB. The utility VDPUTIL from the disk library will enable a TIB program to run in XB.

c. Send me the tape and a disk and I will MAKE it run from disk- 32k ram required! Tools available include Myarc XB, which is an excellent conversion tool, and procedures to shorten programs.
========
PLEASE: Can someone write a little article for beginners on how to set up and use one or both of the database programs Creative Filing System OR PRBase! . There are other powerful programs that need an article or two to enable us all to see what they can do, and how we can make them do it!
========
Q. How do I redefine the CURSOR in a running XB program?
A. The answer to this query is on page 57 of TI*MES Issue 8, and comes from Jim Peterson of Tigercub Software. It REQUIRES 32k ram.

```
100 ! Define character N
with the definition you
require.Now proceed:
110 CALL CHARPAT(N,A$) :: FO
R J=1 TO 16 STEP 2 :: H$=SEG
$(A$,J,2) :: CALL HEXDEC(H$,
D) :: T=T+1 :: H(T)=D :: NEX
T J
120 CALL INIT :: CALL LOAD(8
196,63,248)
130 CALL LOAD(16376,67,85,82
,83,79,82,48,8)
140 CALL LOAD(12288,H(1),H(2
),H(3),H(4),H(5),H(6),H(7),H
(8))
150 CALL LOAD(12296,2,0,3,24
0,2,1,48,0,2,2,0,8,4,32,32,3
6,4,91)
160 CALL LINK("CURSOR")
!program here -------  then:
20000 SUB HEXDEC(H$,D) :: N=
1 :: DEC=0
20001 FOR J=1 TO LEN(H$) ::
A$=SEG$(H$,LEN(H$)-J+1,1) ::
 IF ASC(A$)>58 THEN HT=ASC(A
$)-55 ELSE HT=VAL(A$)
20002 DEC=DEC+N*HT :: N=N*16
 :: NEXT J
20003 IF DEC<=32768 THEN D=D
EC ELSE D=-(65536-DEC)
20004 SUBEND
20005 END
```

========== ===========
If anyone is stuck adventuring, I have a collection of hints and solutions here and may be able to assist- drop me a line WITH SAE describing your problem as fully as possible! and indicate if you want a hint or a step by step guide!
============

TI EDITOR ASSEMBLER BUG:
If you wish to use PIO as an output device for a LIST, you have to put a full stop after it to make it work - eg PIO. - due to an error in the ASSM2 file. The error lies in the DSRLNK routine which the assembler uses - it does not use the same routine that YOUR programs call! The assembler internally uses a modified VSBR routine which returns data to R0 instead of the usual R1, but whoever wrote the internal DSRLNK forget this! Consequently, the assembler looks at R1, and the program uses an incorrect name length when scanning for the device name! The period causes the routine to exit, and "catches" the error.

Want to fix it? If you still have the original ASSM2 file, use a sector editor to look for the code 04 20 AA BE 00 00 D1 C1. It is usually in the 8th sector of the file.

Change that last C1 into a C0 and the Assembler will work as it is SUPPOSED to! No need to add a full stop after PIO now!

Courtesy Danny Michael (Dump and Neatlist author), SHOALS 99ERS.

===============

From Tips from the Tigercub Number 8:

Original:
```
 100 IF X=1 THEN Y=7 ELSE IF X=2 THEN Y=33 ELSE IF X=3 THEN Y=19 ELSE IF X=4
THEN Y=21.
```

Better:
```
 100 Y=VAL(SEG$("07331921",X*2-1,2))
```
=======================

The disk library has a disk called ED AS ONLY (machine code programs which stupidly INSIST on using that module!) which has an odd program called WATOR. I have just located the inspiration!

Quote:"Sharks and fish fight out their ecological battle on the toroidal planet Wa-Tor, discretely simulating stochastic Lotka-Volterra cycles". Yep, it does that! You input various variables and let the sharks and fishes play. You can read all about it in THE ARMCHAIR UNIVERSE by A K Dewdney, published by W H Freeman, for £10.95 in paperback. NOW all I need is someone to translate the programs input prompts from German to English. Volunteer required!

==============

There is ONE disadvantage to using carbon film ribbons on a dot matrix printer- the oil based ink on a fabric ribbon acts as a lubricant for the matrix pins (as well as jamming them up if you use a "cheap" third party ribbon!). So you may have difficulty with sticking pins if you use a carbon film ribbon all the time!).

===============

I tend to cycle round the programs in my collection - it takes quite a while - and from time to time rediscover some interesting programs.

Music disks have not been popular in the disk library, although my son and I are quite fond of them. A POP MUSIC disk from Germany with four (non-pop!) musical items is very much liked, as are the several Sam Moore programs in XB, which are mainly on Amnion disks.

Speaking of music disks, a recent addition is a 77 minute rendition of J C Bach's Sonata for Pianoforte Opus 5 Numbers 1 to 6- on two disks of machine code! Think of the time spent coding it! And no authors name either.

Long long ago, when there were 20 members in the ONLY TI UK Users group, a software company called APEX TRADING was set up by Vince Apps, who also wrote what may have been the first UK published TI Book (a collection of programs). These were early days, and Vince's programs were not exactly classics, but one I heard several complaints about I have just been playing, and find the complaints then heard quite unjustified. I mean- what sort of flight simulator can YOU write in TI Basic??? I think Vince's PILOT program was quite playable. It is certainly retained on MY playlist. (Anyone know what Vince is doing these days?).

Possibly I am the only disk owner to play TI Basic games these days- but there really are several playable games in Basic and ExBas, especially if like me your reflexes are off, and after a days work you want to relax rather than wind up!

Cassette owners- check out our cassette library this quarter, as I have supplied around a hundred (?) of the games I play to Tim, to beef up our collecton. What games do YOU enjoy playing?

==================

DM1000 * CARE *

Your disk library received a copy of files which appeared to be version 4.0 of DM1000 - but without the usual documentation. Very odd! I checked up with the Ottawa Group, who are responsible for this little baby, and it would appear that somehow a "beta test" version of Vn 4.0 got out, complete with bugs. This is what Ottawa said:

"We would like everyone to know that the Ottawa TI-99/4A Users' Group recommends that anyone using DM1000 Vn 3.6 or higher go back to 3.5. It is the most stable version at the moment, and later versions may [ss: nice word: MAY!] contain potentially serious bugs. We apologise to anyone who may have experienced them, and hope that they will enjoy using 3.5 until our next release.

"Version 4.0, for example, is an escaped beta copy and should NOT be used under ANY circumstances. We hope to have a new version (thoroughly beta-tested) ready to go in the fall."[ss: watch for one around Christmas/NewYear]

The group library alas no longer holds a copy of Version 3.5, apart from the amended version 3.5 which is on the Funlweb disk (Vn 4.0 of Funlweb). Therefore if you send for the DM1000 disk itself, to get at the docs, you will receive Version 3.8 of DM1000, which you use at your own risk. You can of course use the Vn 3.5 on the Funlweb disk.

It is problems like this that cause your disk library to always try to obtain programs from source!

=======

David Caron of Ottawa reminds me of these tips for XB users...

1. In order to quickly tell the difference between I and 1 and between O and 0 when keying a program in - a source of frequent errors! - first key this in, in immediate (command) mode:

```
    CALL COLOR(3,16,1) :: ACCEPT AT(1,1):A$
```

... when you press enter, the cursor will appear at top left.
... Press CLEAR [FCTN 4] and you can begin to type in your program, and the numbers 0 to 7 will really stand out!

2. To REALLY annoy someone, with a VERY long sound in your program, instead of the usual meagre 4 seconds or so, and without having to add more CALL SOUNDs, add these lines to your program-and turn the noise (sorry, sound) on with a GOSUB 1000:

```
      1000 ! NOISE
      1001 ON ERROR 1005
      1002 CALL SOUND(-4150,110,0)
      1003 CALL SOUND(-9999,110,1)
      1004 GOTO 1006
      1005 RETURN NEXT
      1006 RETURN
```

(May not function on all XB modules... tell me if it doesn't on yours, and quote the serial number on the module and the error message!).

=======

For some odd reason I do my computing near to a scented amphora, with lemon oil gently scenting the air.... obviously way ahead of everyone else, as New Scientist just reports that operating a keyboard in the vicinity of lemon oil reduces keyboard errors, presumably by reducing fatigue??? Thought you'd be interested...

=======

After a little experimentation with member K F Hughes, I can report that members who only have cassette players but also a 32k ram expansion (TI standalone or home built) CAN load machine code programs from tape, and we are able to supply several items this way. Tape owners with 32k, please contact me for the time being...

=======

TI BITS * Number 4 * By Jim Swedlow

[This article originally appeared in the User Group of Orange County, California ROM]

{ Excerpted by Stephen Shaw June 1988}

PROGRAMMING TIP

Suppose you are writing a program that does a great deal of printing. There is a bug somewhere in the middle of the printing instructions (ss: ... causing a I/O Error message}.

Every time you try and find it, however, you must wait while your printer wastes a lot of paper getting to the problem. What to do? If your printer is PIO, try substituting RS232.BA=9600. Unlike the parallel port, the serial port does not wait for a ready signal to return from your printer. So all of your print instructions will go out thru the RS232 port into thin air until you find your problem. Setting the baud rate at 9600 speeds things up (if you don't specify a rate, your TI will use 300 - much slower).

MORE QUOTES

"Only those who attempt the absurd achieve the impossible."
---Anon

AN INTRODUCTION TO PRINTERS

If you are thinking about buying a printer, beware. Your choices are many as are the pitfalls.

First, you will need some things other than a printer. You need an RS232 card (stand alone or one for your P Box) and a cable. Most printers work with the parallel port using a standard TI-Printer cable (available from the houses that still support the 4A - Tenex, Tex-Comp, etc).

{ss:It is also possible to purchase stand alone parallel interfaces which come complete with cable, and there are also a word processing module and a spread sheet module which you can buy which have a parallel cable as part of the module.}

But which printer to buy? Epson? Star? Gorilla? Tandy? What kind? Dot matrix? Daisy Wheel?

First, lets look at the two basic types: daisy wheel and dot matrix (the others are probably out of your price range). A dot matrix printer is five to ten times faster and much more versatile. A daisy wheel gives you letter quality print while the dot matrix gives draft (poor) and 'near' letter quality (better). A tractor feed usually comes with a dot matrix printer but can be an extra cost item with a daisy wheel printer.

If 90% of your work is correspondance and you need top quality in its visual presentation, a daisy wheel is probably for you. Otherwise, for listing programs and all the other things that a printer can do, a dot matrix printer is the better choice.

{ss: A dot matrix printer is essential if you wish to print any Graphics!}

Having narrowed the field, you still have to pick between the many models on the market. There are no official standards in the world of printers for command structures (the codes your computer sends to the printer to tell it what to do). About the only two codes that are close to universal are Carriage Return and Line Feed. After that, anything can mean anything.

There are two 'de-facto' standards. The first is IBM. When big blue made a printer for its PC, it used a character set and command structure which differed from ASCII and just about every printer on the market. Alas, what will work with an IBM PC will NOT work on most 4A software , so IBM compatibility is useless (unless you plan to defect).

The other quasi-standard is Epson. These folks developed a rather comprehensive instruction set (including graphics protocols) that some other manufactures and many software manufacturers followed. The TI impact printer is actually a bottom of the line Epson MX80. Most of the graphics programs for the TI will work with Epsons. Some of them support other printers, others do not.

A number of manufacturers make printers that follow Epson commands. Most Star (Gemini 10X, SG10, etc) and Panasonics do while the Axiom, Tandy and Banana printers do not.

Here are some suggestions to help you choose. First, see what your friends have and what they think of it. Then, in the store, have the salesman show you the draft and near letter quality print fonts. Note how long it takes to print a page (200 cps - characters per second - means different things depending on who is writing the advertising, I mean specs). Look for true decenders (is the loop below the 'g' below the line?) and the difference between the zero (0) and the letter (O). Make sure you can return it if it doesn't workout.

{ss: On most Epson printers you can select to have your number zero crossed or uncrossed}

Plan to spend at least £180 (if you are buying a new printer). Any of the bargains below that normally do not have the features you will need.

{ss: and in many cases very cheap printers are obsolete or for some other reason you cannot easily buy replacement ribbons, or if applicable, print wheels or thermal paper. You are warned! All Epson MX80/FX80/FX85 printers use an identical easily found ribbon!}

My printer? A Star Gemini 10X. Its about 85% Epson compatible and has been a faithful companion.

{ss: MY faithful companion is an Epson FX80, now over 5 years old, and look how many words I have written!}

ANOTHER PROGRAMMING HINT

When working on a program, you save it to disk often just in case your system locks up, etc. To save time, use a working name of <A> for these frequent saves. This saves up to nine key strokes. Also, if you have a load program that reads the disk directory, your working program will be at the top of the list.

{ss: For greatest protection, save to TWO different file names: After making a little alteration to the nearly finished program, you save to file A, and in the middle of the save, you crash.... and file A is no more, so you better have a file B on there....}

Enjoy!

[Contents of File TI04]

```
X   X   BBBB
 X X    B   B
  X     BBBB      By
 X X    B   B     Jim
X   X   BBBB      Swedlow
```

[This article originally appeared in the User Group of Orange County, California ROM]

{Excerpted by Stephen Shaw June 1988}

IF THEN ELSE - AGAIN: Another thing I found in the 'Teach Yourself XB' tutorial is how XB matches ELSE's with IF's.

Each ELSE is paired with the last unmatched IF. For example:

```
IF A THEN B :: IF C THEN D ELSE E ELSE F
```

In words: If A is true, do B and then test C. If C is true, do D. If C is false, do E. If A is false, do F.

DISK MENU PROGRAM: (TI*MES Issue 20, Page 14): There was an error in this program, in line 210: the HCHAR parameters should have been (5,1,32,32*20).

WHEN ALL ELSE FAILS . . .
TRACE and UNTRACE are powerful debugging tools. The other day I was looking something up in the XB book and discovered that these can be used as statements in a program line. This makes them all the more helpful in catching that bug!

IMAGE and USING
XB left justifies all printed and displayed strings and numbers. While this is correct for strings, numbers should be lined up by the decimal point. One hard way to fix this is to turn your numbers into strings and add leading spaces. PRINT USING is much easier.

Enter and run this program:

```
100 FOR I=9 TO 10 STEP .1
110 PRINT USING "##  ##.#  #.##"
    :I:I,I :: NEXT I
```

Note how the first column prints the number rounded to the nearest whole number while the other two display decimals. The string of asterisks shows you that the number was too big for the space alloted.

Only the number on the screen is rounded -- the number in RAM is NOT rounded.

The statement after USING can be a string, a string variable or a line number that refers to an IMAGE statement. Examples:

```
10 A$="The answer is ###.##"
   :: B$="John" :: C$="Dear"
20 IMAGE ## + ## = ###
30 PRINT USING A$:2.45678
40 DISPLAY AT(1,1):USING 20:
   14,19,14+19
50 PRINT #1, USING "$###.##":23.1
60 PRINT USING C$&"#####":B$
```

For more details, look in the XB book. Remember, when all else fails . . . .

WD 40

Recently Ramon mentioned using WD 40 to clean a disk drive. WD 40 is an excellent solvent but a lously lubricant as it attracts dirt. Further it does bad things to plastic. So use it to clean up gunk on metal but be sure and use a good quality oil to lubricate.

SUB PROGRAMS

XB has four (or more) ways of writing utility routines: sub routines, sub programs, assembly language routines and DEFinitions. Each of these has pros and cons. Let's look at sub programs.

The best way to describe a sub program is to explain where it differs from a subroutine:

-- Sub routines are called by line number while sub programs are CALLed by name.

-- Sub routines use the same variables as the main program. In subprograms , variables are entirley different unless noted in the CALL statement.

-- Sub routines can be anywhere in the program while subprograms must be at the end.

Since you call sub programs by name, you do not lose their identity when you RESquence. You lose the power of ON GOSUB and passing varialbes can be tricky if quite a few are involved.

Enter and run this sample program:

```
  10 A,B=10 :: PRINT A;B;C ::CALL TEST
  20 CALL NEWS(B) :: PRINT A;B;C
  30 SUB TEST :: A=5 :: PRINT A;B;C
     :: SUBEND
  40 SUB NEWS(C) :: PRINT A;B;C :: C=4
     :: SUBEND
```

You should get this output:

```
 10 10  0
  5  0  0
  0  0 10
 10  4  0
```

If you think this thru, you will see that there are NINE variables in this program: A, B and C in the program; A, B and C in TEST; and, A, B and C in NEWS.

When you called NEWS(B) the value of B was passed to NEWS and processed as C in that sub program. Play with this program (including inserting BREAK statements) to test this out.

ORACLE

This program is loosely based on one that appeared in '99er years ago. Both programs were designed to demonstrate sub programs. Speech has been added to this version.

The CALL PEEK in line 160 works on my computer but may not work on yours -- if you have a problem, just delete it. If you don't have speech, change the CALL SAY's to PRINT's.

Various formats for sub progams are shown. The rules are in the XB book, especially for passing variables between the main program and the sub program. Note that SUB name must start a line and SUBEND must end a line.

A disclaimer: all answers are based on random numbers. If you think the answers fit, well, that's just random chance, right? A computer can't give good answers, can it?

Enjoy! (Program listing follows)

```
100 ! ORACLE
110 ! VERSION XB.2.1
120 ! 08 MAR 85
130 ! BY JIM SWEDLOW
140 !
150 DISPLAY AT(10,4)ERASE AL
L BEEP:"!! I AM  THE ORACLE
!!" :: CALL DELAY :: RANDOMI
ZE
160 CALL INIT :: CALL PEEK(-
28672,I):: IF I=0 THEN DISPL
AY AT(20,1):"I cannot operat
e without the    Speech Synt
hesizer!" :: STOP
170 DISPLAY AT(15,1):"   I a
nswer all questions": : :"As
k qustions with  YES or NOan
swers -- When you are donepr
ess ENTER."
180 CALL DELAY :: DISPLAY AT
(24,1):"   PRESS ANY KEY TO
BEGIN"
190 CALL KEY(0,I,S):: IF S=0
 THEN 190 ELSE CALL CLEAR
200 PRINT : :"WHAT IS YOUR Q
UESTION?" :: LINPUT Q$ :: IF
 Q$="" THEN 220
210 CALL DELAY :: CALL REPLY
(Q$):: CALL DELAY :: GOTO 20
0
220 DISPLAY AT(10,1)ERASE AL
L:"THANK YOU FOR CONSULTING"
: : : : : : : :"    !! THE ORA
CLE !!" :: CALL DELAY :: STO
P
230 !
240 SUB DELAY :: FOR I=1 TO
200 :: NEXT I :: SUBEND
250 !
260 SUB REPLY(A$)
270 A$=SEG$(A$,1,2):: IF A$=
"WH" OR A$="HO" THEN CALL OT
HER ELSE CALL YESNO
280 SUBEND
290 !
300 SUB YESNO
310 ON INT(10*RND)+1 GOTO 32
0,330,320,340,350,350,360,37
0,380,390
320 CALL SAY("YES"):: SUBEXI
T
330 CALL SAY("I THINK SO")::
 SUBEXIT
340 CALL SAY("LOOKS POSITIVE
"):: SUBEXIT
350 CALL OTHER :: SUBEXIT
360 CALL SAY("LOOKS NEGATIVE
"):: SUBEXIT
370 CALL SAY("I DO NOT THINK
 SO"):: SUBEXIT
380 CALL SAY("NO WAY"):: SUB
EXIT
390 CALL SAY("NO"):: SUBEND
400 !
410 SUB OTHER
420 ON INT(10*RND)+1 GOTO 43
0,440,450,460,470,480,490,50
0,510,520
430 CALL SAY("I CAN NOT TELL
 YOU THAT"):: SUBEXIT
440 CALL SAY("SAY THAT A DIF
FERENT WAY"):: SUBEXIT
450 CALL SAY("YOU DO NOT   iN
T TO KNOW"):: SUBEXIT
460 CALL SAY("I D  NOT KNOW"
):: SUBEXIT
470 CALL    Y("I AM NOT SUPPO
SED TO SAY"):: SUBEXIT
480 CALL SAY("I WILL NOT TEL
L YOU"):: SUBEXIT
490 CALL SAY("I CAN ONLY GUE
SS"):: SUBEXIT
500 CALL SAY("I CAN NOT ANSW
ER THAT"):: SUBEXIT
510 CALL SAY("I DO NOT REMEM
BER"):: SUBEXIT
520 CALL SAY("TRY SOME THING
 ELSE"):: SUBEND
530 END
```

[Excerpted from files XB06 to XB08 by ss]

LISTING UTILITIES

There are many ways of producing program listings, and many ways of manipulating programs. This little article is to show you one program, which has been manipulated with three others, so you will see the original and three variations. Not shown here is the 28-column format, produced by Tony McGovern's COLIST program, as that has been used many times in this magazine already.

COLIST, UNBASHER, and NEATLIST are all available from the user group disk library, while SMASH is a commercial program available from Tenex.

First the original program. I am only showing a central portion of the program, rather than take up too much magazine space!

```
1440   FOR X1=1 TO 4                  1580  W=W+1
1460   R(X1)=ASC(SEG$(P$,X1,1))       1600  R(J)=0
1480   NEXT XI                        1620  GOTO 1660
1500   W=0                            1640  NEXT J
1520   FOR I=1 TO 4                   1660  NEXT I
1540   FOR J=1 TO 4                   1680  W=W-B
1560   IF G(I)<>R(J)THEN 1640         1700  RETURN
                -------> 
```

Once this program has been SMASHed, these lines look like this:

```
1440 FOR X1=1 TO 4 :: R(X1)=ASC(SEG$(P$,X1,1)):: NEXT X1 :: W=0 :: FOR I=1 TO 4
 :: FOR J=1 TO 4 :: IF G(I)<>R(J)THEN 1640
1580 W=W+1 :: R(J)=0 :: GOTO 1660
1640 NEXT J
1660 NEXT I :: W=W-B :: RETURN
```

Which can be made a little more meaningful by running it through NEATLIST:

```
01440 FOR X1 = 1 TO 4 ::
         R(X1) = ASC(SEG$(P$ , X1 , 1)) ::
         NEXT X1 ::
         W = 0 ::
         FOR I = 1 TO 4 ::
         FOR J = 1 TO 4 ::
         IF G(I) < > R(J) THEN 1640
01580 W = W + 1 ::
         R(J) = 0 ::
         GOTO 1660
01640 NEXT J
01660 NEXT I
         W = W - B ::
         RETURN
```

```
VARIABLES USED IN MAIN PROGRAM
(in lines 1440 to 1700 only)
G                       01440
I                       01440 01660
J                       01440 01580 01640
P$                      01440
R                       01440
W                       01440
X1                      01440
```

(The variable listing is available as an option with the listing, or on its own).

We can also re-split the SMASHed program by using UNBASHER, which produces the following. Note that the output has not been resequenced, but of course it could have been:

```
1440   FOR X1=1 TO 4
1441   R(X1)=ASC(SEG$(P$,X1,1))
1442   NEXT X1
1443   W=0
1444   FOR I=1 TO 4
1445   FOR J=1 TO 4
1446   IF G(I)<>R(J)THEN 1640
1580   W=W+1
1581   R(J)=0
1582   GOTO 1660
1640   NEXT J
1660   NEXT I
1661   W=W-B
1662   RETURN
```

You may not have a lot of control over the format of a program you purchase or receive from elsewhere, such as the library, but the utilities are available to change the format to something which may be easier to read/debug, or which perhaps occupies a little less memory space and runs a trifle faster.

===

ENHANCED BASIC
Part Two

Or What You Can Do when you have the Personal Record Keeping or Statistics modules inserted into your console, or, if loaded from disk, if your console thinks they are inserted.

Last issue we dealt with the really easy bit, the equivalent calls to ACCEPT or DISPLAY data.

This issue we shall dive into the mysteries of CALL L (load), CALL P (partition), CALL H (header), and CALL G (getput).

The best way to introduce these perhaps is with a working program, then in the next issue we can get down to a little more detail!

Program first, then discussion. This program will use an already created PRK data file and in the TI Basic environment will read that data and display it on the screen. It will demonstrate several of the calls in a fairly simple manner, so you can get used to the idea of using PRK data in a Basic program!

If you have a disk system, free up memory by typing:

```
CALL FILES(1)   [ENTER]
NEW     [ENTER]
```

before you load and run this program!

As this program occupies some VDP memory, you may find that a few PRK data files will be too long to load. Try deleting some records to make the data file fit.

Remarks will be in lower case between the program lines!

BEFORE LOADING THE PROGRAM you must allocate an area of VDP memory for the data, by using the partition command, in command mode, thus:

```
CALL P(10900)    [ENTER]
NEW        [ENTER]
```

The program follows with lower case comments:

First, load the PRK file from disk or cassette:

```
 100 CALL L("CS1",Y)
         or
 100 CALL L("DSK1.FILENAME")
       as you wish.
 110 IF Y=0 THEN 500
     Check to see if the file is loaded. No?  There was an error,
     else carry on.
     How many fields in each record? Lets   find out, where F will
     return number   of fields.
 120 CALL H(1,5,0,F)
 130 CALL CLEAR
     Display the result.
 140 CALL D(1,1,28,"# of FIELDS:"&STR$(F))
     Then we need to know how many records there are in the data file —
     think of a record as a page:
 150 CALL H(1,6,0,R)
     Display the result.
 160 CALL D(2,1,28,"# of RECORDS:"&STR$(R)
Now let's loop through each record and display it on the screen,
 170 FOR RCD=1 TO R
     taking each field in turn.
 180 FOR FLD=1 TO F
But does the field contain a number or a string? Let's find out
 190 CALL H(1,10,FLD,TYPE)
 200 IF TYPE=1 THEN 240
     If type is 0, data is a number.
     Get the data from field FLD of record RCD,
 210 CALL G(1,RCD,FLD,Z,RD)
     print the result on screen,
 220 CALL D(2+FLD,1,28,RD)
     and jump over string section.
 230 GOTO 260
     This is string section:
 240 CALL G(1,RCD,FLD,Z,RD$)
 250 CALL D(2+FLD,1,28,RD$)
That's one field dealt with, on to the next.
 260 NEXT FLD
     Now all fields in the record are on screen. Let's give
     the record number and a pause.
 270 CALL D(23,18,10,"RECORD:"&STR$(RCD))
 280 CALL D(24,1,28,"PRESS ENTER FOR NEXT")
 290 CALL A(24,20,1,RCD,RD$)
     And on to the next record
 300 NEXT RCD
 310 PRINT "NO MORE"
 320 GOTO 320
 500 CALL CLEAR
 510 PRINT "LOAD ERROR"
 520 GOTO 520
 530 END
```

CALL P sets aside an area of memory for the data to be stored in, and must of course be sufficient to hold the data, but, for Basic use, we must also have enough room for our Basic program. After using CALL FILES(1), disk users have a maximum of 12768 bytes of VDP RAM free to split between the Basic program and data area, while cassette users have 13820 bytes available (see how much memory a disk system eats up!). By using 10900 for our data, we have left even disk users with at least 1868 bytes for our little program!

CALL L just loads the data file into the partitioned area, the return variable following the device/file name is a 0 (zero) if an error occurs, for instance, the files is not on the disk, the data is too large for the partition, or other I/O errors. The device/filename may be a string variable if you wish,
e.g.: CALL L(A$,A)

CALL H deals with the "header" information — the database specification you create when using PRK/Statistics: what sort of data is held in each field and what name have we given the field?

As used in the above example (there are many other uses!):

```
CALL H(1,5,0,F)
CALL H(1,6,0,R)
CALL H(1,10,FLD,TYPE)
```

The first digit (1) is a READ instruction. We would use a 0 to write.
The fourth digit is a variable placed in or obtained from the header data.
The second digit (5, 6, or 10 above) is the header item number, from 1 to 14, where 5 is the number of fields per record, and 6 is the number of records. 10 is the type of field, returning to the fourth item, a numeric variable, a 1 for characters (string), a 2 for integer, a 3 for decimal, and a 4 for scientific notation.
We shall deal with the other 11 header items, and with writing to the header, in a later article.

CALL G deals with the actual data we have stored, and as used above (again there are other uses):

```
CALL G(1,RCD,FLD,Z,RD)
CALL G(1,RCD,FLD,Z,RD$)
```

Again the first item (1) is a READ instruction, with 0 used to write.
The second and third items identify the record and field number that we are going to read/write, while the final item is the variable that the contents of that particular record/field will be placed in for us to use. Note that we must use a string variable for characters!
That fourth item (Z) is a numeric variable, which can be used to indicate a blank field. It takes a value 0 if data is found, and 1 if data is missing, that is, not entered!
When using CALL G to write, the fourth parameter as above is not used, you just go on to the value you are writing, for instance:
CALL G(0,RCD,FLD,"STRING INPUT")
That fully covers CALL G, apart from a sample of how to write!

CALL S will Save the data area to cassette/disk, very easily indeed:
CALL S(DEVICE$,VARIABLE)
with variable again indicating success or failure!

That leaves us with a great deal to look at with CALL H, and also we need to look at writing data, and (perhaps!) maybe even creating a data file without using the PRK create file routine.Until later...

(Is anyone reading this please?! Is this the right level/approach for you?)

==

GRAPHICS COMPATABIL1TY

This article has been prompted by a very odd chart of the various graphics programs for the TI which I came across in a US newsletter – odd because at the end of the day it failed to tell you very much and was decidedly biased!
This article also follows – in a way – from the discussion of various formats of disk file.
Each type of file is referred to by means of a short abbreviation, details of which are given in the first section below:

1. List of formats

TI-ARTIST..........	Fonts (_P files, referred to later as TIAF).
	Pictures (_P and _C files, referred to as TIAP)
	Slides (_S files, TIAS)
	Instances (_I files, TIAI)
GRAPHX	Clipart, including fonts (GC)
	Pictures (GP)

continued--->

CSGD	Pictures (/DT files, CP)
	Graphics (/GR files, CG)
	Fonts usual (/CH files, CF) care: see note at end!!
	Fonts — DocuPrinter (/DP files, CD)
	Labels (/LB files, CL)
	Letter headings (/IL files, CH)
JOY PAINT.........	Pictures (JP)
	Compressed pictures (JC)
PICASSO	Pictures (PP)
	Fonts (PF)
	Icons (PI)
BITMAC	Pictures (BP)
DRAW 'N PLOT —	Pictures (DP)
DRAW-A-BIT 1 —	Pictures (DAB1)
DRAW-A-BIT2 —	Pictures (DAB2)
MAX/RLE	Pictures: D/V 80 files or D/F128 files (MP)
PAINT 'N PRINT Module —	PNPP.

MAX/RLE and Picasso are available from the disk library.

Note: CSGD uses two different sets of /CH files. The font editor creates one set of /CH files, which then have to be converted to another type of /CH file for use. The /CH files referred to here are always the converted files. The conversion program is on CSGD Volume 1.

2. MUTUALITY

This section indicates the types of file each graphics program can use from the above list, WITHOUT using an external conversion utility. The ability to both save and load can be assumed unless otherwise noted:

MAX/RLE	TIAP, GP, MP.
TI-ARTIST	TIAP, TIAF, TIAS, TIAI, DAB1, DAB2, DP, GP.
GRAPHX	GC, GP.
CSGD 1 AND 2 ...	CP, CF, CG.
CSGD 3	CF, CG, CH, CL and (CD -load only).
PICASSO	PP, PI, PF, TIAP. Can also load a TI-Writer text file.
JOY PAINT	JP.
JOY PAINT PAL 2	JP, TIAP, JC. Can also load GP, DP.

Where more than one type is listed in the above section, conversions are possible as part of the main program, which is usually much faster.

3. GRAPHICS UTILITIES

These include external (e.g., separately loaded) conversion routines on the main graphics disks:

THE PRINTER'S APPRENTICE:	Uses its own picture and font formats, can also use TIAP.
TPA TOOLBOX:	Uses TPA fonts and graphics, plus can convert into TPA format the following: TIAI, TIAF, TIAP, CF.
PRINT WIZARD:	Creates its own format from TIAI and TIAF.
FONT WRITER 2: . . .	Uses, in various utilities, TIAF, TIAI, TIAP, CF, CG, GP. Can convert: CG to TIAI, CP to TIAI, TIAI to CG, and TIAI to CP.
PICASSO:	Can convert an XB font to PF, or load a PF into an XB program. Convert BP to PP. Make use of CF and CG files.
EXTENDED GRAPHICS PACKAGE:	Requires Paint 'n Print module.

```
CSGD 1: .....  .....   ..  Can convert an XB screen into CP.
ARTIST EXTRAS (TEXAMENTS): Can convert: CF or CD to TIAF, CG to TIAI, and CP
                           to TIAI. Allows Super Sketch to be used as an input
                           device for TI-Artist.
ARTCONVERT (TRIO+): ....   Can convert TIAI and TIAF to TI-Writer graphics.
ARTIST ENLARGER (ASGARD):  Works with TIAF and TIAI.
GRAPHICS EXPANDER and Bigtype (Genial): Works with TIAF, TIAP, and TIAI.
GRAPHICS LISTER (NAMELOC). TIAI.

PICASSO UTILITIES (ASGARD): Ad description fails to indicate what this does.
DISPLAY MASTER (INSCEBOT): TIAP.
CSGD CATALOGER (TEXAMENTS): CG. CF, TIAF, TIAI.
GRAPHX SLIDE SHOW (ASGARD): GP.
DESIGNER LABELS (TEXAMENTS): TIAI.

EXTENDED BUSINESS GRAPHS (Great Lakes): JP.
CHART MAKER II (QS99):     DP.
CALENDAR MAKER 99(ASGARD): TIAI.

Picture it (Merritt) (disk library):TIAI to Banner, XB, and TI-Writer.
Graphic Labeler (disk library): CG.
JBM103 (disk library): Enables graphics to be loaded/saved to/from Extended
                       Basic bit-mapped screens in TIAP format.
UTIL12 (disk library): Has a utility to convert from TIAI to Extended Basic
                       PROGRAM format - MERGE file, or listing to disk or printer.
UTIL 7 (disk library)L Has a utility to convert TIAI to TI-Writer graphics.
UTIL17 (disk library): Has a utility called XBGC to convert a segment
                       (5 x 5 chars) of a GP to CG, and a utility to convert
                       CG to TIAI and/or Extended Basic MERGE file.
TASS (disk library): TIAP, GP, DAB2. Slide show.
COMIC 1 (disk Library): A utility which enables you to create a machine code
                        animation sequence from up to 100 TIA Pictures (TIAP).
                        The animation speed is adjustable as the program runs,
                        and can be very fast indeed.
Myarc Utilities (disk library): TIAP and GP to load into Myarc XB program.
```

Programs in MICROpendium. Back issues available to MICROpendium subscribers:

1987 Feb. Convert TIAP to/from Myarc XB.
1987 July. Rotate TIAI.
1987 Oct. Print utility. TIAI.

PICASSO, and PICTURE IT are copyright. Other disk library programs are faireware. All other programs are copyright commercial programs.

The de facto standard has been set by TI-Artist - only graphics programs released before TI-Artist lack TIA capabilities, apart from CSGD, although external utilities have been created to remedy that!

As far as printers go, all these work with Epson FX series printers or any printer which follows Epson commands. The usual commands used are:

```
ESC *     (8-pin bit image mode )
ESC K     (480 dot 8-pin mode )
ESC L     (960 dot 8-pin mode)
ESC Z     (1920 dot 8-pin mode)
ESC A n   (Line spacing in n/72 inch)
ESC 1 n   (Left margin setting)
```

A few programs allow Gemini printers to be used, but Gemini used two incompatible codes in their printers, and Gemini owners often report problems. A very few programs will support other printer codes.

The vast majority of the programs listed above remain available. Not a bad choice at all for an orphan computer, whose manufacturers left it with a Video-Graphs module which compares very badly with the above.

I have deliberately omitted a few simple programs such as Norton Graphics Package, and an input device, Super Sketch – but note that Super Sketch can be interfaced to TI-Artist. Any other omissions are due entirely to my ignorance of the products involved. Or in the case of very new programs, news was not with me when this was printed.

==

TI-WRITER DATABASE
TI-Writer can function as quite an effective database, if you use FS to locate a specific item. You can even use M to put your items into order.

The program which follows will enable you to use each line of TI-Writer as a separate record, divided into up to 9 fields, and with up to 400 records. Not bad, eh! Quite sufficient for simple database purposes.

This program is an Extended Basic sorting routine, which will sort the records by any one field.

The fields are identified in line 1 of the TI-Writer file, by numbers located in the left-most column of each field. Numbers may be zero filled, or right justified, as follows:

```
1            2            3            4            5
TI*MES       JAN-MAR 88   43           PILOT 99     ARTO HEINO
TI*MES       JAN-MAR 88    8           DISK DRIVES  STEPHEN SHAW
.....
```

Save the D/V 80 file using SF as usual. This program will strip the tab records out for you!

Use of TI-Writer as a database is of value as it is familiar, easy to use, and easily edited!

This program comes from Computer Bridge, Volume 5 No. 11, Nov 1986, and was written by

```
100 REM PRESCAN
110 GOTO 120 :: D$,X,Y,K,S,Z
,FN,FS,FL,L,S$,H,J,B$ :: CAL
L KEY :: CALL SOUND :: CALL
HCHAR
120 DISPLAY AT(4,2)ERASE ALL
:" See Instructions? Y/N" ::
 DIM A$(400),F(10)
130 !@P-
140 ACCEPT AT(5,12)VALIDATE(
"YN"):D$ :: CALL CLEAR :: IF
 D$="N" THEN 230
150 PRINT "This program was
written to sort files create
d with     TI-Writer. The fi
le can be as long as 400 li
nes."
160 PRINT "The first line of
 the file must contain the
field      numbers, like thi
s:"
170 PRINT "1 2 3 <-first
line":"102 NJ Paul Peters":"
314 MO Carol Corrina":"622 A
L Hu Noes":
```

---> --->

```
180 PRINT "You may have up t
o 9 fields but they must all
 fit on oneline."::: INPUT "
Press <ENTER>":D$ :: CALL CL
EAR
190 PRINT "The lines can be
up to the full 80 character
s long thatTI-Writer support
 s. This   program will only
 sort one field at a time."
200 PRINT :"The sort is alwa
ys":"ascending order, with t
he     lowest value first."
210 PRINT "The file can be u
sed by     TI-Writer after t
his programis through with i
t.":
220 INPUT "Press <ENTER>":D$
 :: CALL CLEAR
230 DISPLAY AT(12,1):"Input
device and file name:":"DSK
"
240 ACCEPT AT(13,5):D$ :: IF
 D$="" THEN 240 ELSE OPEN #1
:"DSK"&D$,DISPLAY,VARIABLE 8
0,INPUT
```

MORE---> --->

```
250 CALL CLEAR
260 REM LOOK FOR 1ST LINE
270 LINPUT #1:A$(0) :: IF EO
F(1) THEN 610 ELSE IF A$(0)=
" " THEN 270
280 FOR X=1 TO 400 :: LINPUT
 #1:A$(X) :: IF EOF(1) THEN
310
290 NEXT X
300 REM F()=field's position
s (count down 2 from eof():l
ast 2 lines are tabs)
310 X=X-2 :: CLOSE #1 :: FOR
 Y=1 TO 9 :: F(Y)=POS(A$(0),
STR$(Y),1) :: IF F(Y)=0 THEN
 330
320 NEXT Y
330 F(Y)=80 :: Y=Y-1 :: IF Y
 THEN 350
340 PRINT "Can't find field
marker":"This was found inst
ead:":A$(0) :: STOP
350 DISPLAY AT(10,5)ERASE AL
L:"Press:":"1. To Sort":"2.
To Save to disk":"3. To QUIT
"
360 CALL KEY(3,K,S) :: IF (K
<49)+(K>51) THEN 360 ELSE CA
LL SOUND(-20,880,0) :: ON K-
48 GOTO 380,540,580
370 REM GET FIELD TO SORT
380 DISPLAY AT(1,3)ERASE ALL
:"<choose field to sort>"
390 FOR Z=1 TO Y :: DISPLAY
AT(Z*2,1):"Field No:";Z;" ";
SEG$(A$(1),F(Z),F(Z+1)-F(Z))
:: NEXT Z
400 CALL HCHAR(20,1,30) :: C
ALL KEY(3,K,S) :: FN=0 :: CA
LL HCHAR(20,1,32)
410 IF (K<49)+(K>48+Y) THEN
400 ELSE FN=K+48
420 REM SORTING
    FN=FIELD NUMBER
430 REM
    FS=FIELDS START
    FL=FIELDS LENGTH
440 DISPLAY AT(12,5)ERASE AL
L:" Sorting..." :: FS=F(FN)
:: FL=F(FN+1)-FS :: K=X :: L
=INT(X/2)+1
450 IF L<>1 THEN L=L-1 :: S$
=A$(L) :: GOTO 470
460 S$=A$(K) :: A$(K)=A$(1)
:: K=K-1 :: IF K<1 THEN A$(H
)=S$ :: GOTO 350
470 J=L
480 H=J :: J=J+J :: IF J>K T
HEN A$(H)=S$ :: GOTO 450
490 IF J>=K THEN 510
500 IF SEG$(A$(J),FS,FL)<SEG
$(A$(J+1),FS,FL) THEN J=J+1
510 IF SEG$(S$,FS,FL)>=SEG$(
A$(J),FS,FL) THEN A$(H)=S$ :
: GOTO 450
520 A$(H)=A$(J) :: GOTO 480
530 REM SAVE ON DISK INC TAB
540 DISPLAY AT(12,1)ERASE AL
L:"Device and filename":"  D
SK";D$
550 ACCEPT AT(13,6)SIZE(-12)
:D$ :: IF D$="" THEN 240 ELS
E OPEN #1:"DSK"&D$,DISPLAY,V
ARIABLE 80,OUTPUT
560 FOR Y=0 TO X+2 :: PRINT
#1:A$(Y) :: NEXT Y :: CLOSE
#1 :: GOTO 350
570 REM QUIT
580 DISPLAY AT(12,1)ERASE AL
L:"<ARE YOU SURE? Y/N>"
590 CALL KEY(3,K,S) :: IF S<
>1 THEN 590
600 IF K=89 THEN STOP ELSE I
F K=78 THEN 350 ELSE 590
610 DISPLAY AT(22,1):"File n
ot properly organised":" <pr
ess enter>" :: INPUT D$ :: G
OTO 230
620 END
```

RAMBLES POST SCRIPT...

Just received...

FUNLWEB Version 4.1, dated 30.5.88, two disks as usual. Wow. Phew. Configuring the system disk is now highly delightful. Worked first time for me, this one. Lovely. Superb. Excellent. Hmm, run out of superlatives. Another triumph from Tony and Will McGovern. Very highly recommended. DM1000 Vn 3.5 remains on the disks but has been substantially reworked by Tony, not that its very visible, just a little more reliable.

My order from TENEX arrived, on 18th June 1988. Two backordered items only, not bad, but the printer cable "guaranteed to work" didn't. Watch out for further reports on that in next issue!

One item I was very interested in - the most costly item in my order indeed - was a 1987 module from Databiotics called TI PLANNER, a spreadsheet in a module which requires NO extra peripherals. Just a console. (Module=US$27).

It is perhaps to spreadsheets what Personal Record Keeping is to databases - perhaps not very impressive to the "power user", but distinctly usable, simple to grasp and use, and requiring nothing fancy.

The Shaw household, never having got to grips with Multiplan, WILL be using THIS small spreadsheet. Sheets can be saved to disk OR cassette! And if you dont have a printer interface, there is a separate version of the module available which has a parallel printer cable trailing out the back of it! Wow.

It does not have move or copy, can't print part of a sheet, has fixed width columns (12 chars wide), has limited formulas (12 chars!) and only a small size- three choices for 32k owners and 3 smaller choices for non-32k owners. But quite adequate for anything I'm likely to use. Default 32k size is 50x50 cells. The '^' function is bugged, do not use it! And calculations are as accurate as the display so if you select no decimal places, be careful with your formulae! I like this product and will use it. Need I say more? *

Less happy is THE BASIC BOOK which I thought might be an update on the excellent book by Lien. No way. The TI99/4 (no A even though it was published in 83) entries are more than half incorrect, and only relate to TI Basic even though in 83 the majority used ExBas! How often have you used the well known TI Command: GO GCHAR? or CALL HCAR or CALL JOYSTK? This book is worth less than the value of its paper, pulped, which it should be. A real gem from publishers McGraw-Hill.

Better books from SAMS (circuit diagrams and photos) and I now have the Bunyard technical book, advertised at US$25 in our January issue. It is a very readable book, with many comments on differences between consoles, making it good reading both for the hardware fiend AND the assembly language programmer, as it points out some areas where code may work on one console but not another!

Scott Adams hint book also in- $8 only - and it IS only hints! NO maps, no solutions. Might still offer a little help though. Seventeen adventures which seem to be available to us, plus QUESTPROBE NO3(?). (2022: = Buckaroo Banzai)

I also have TOTAL FILER, which at first glance has rather restricting documentation which seems to hide the programs talents under a ton of something. It looks interesting but deserves research so more in next issue!

Stephen Shaw

* 2022 update- I never did actually use TI Planner.

BASIC STATEMENTS — IF ... THEN ... ELSE

In the last TI*MES we met the IF ... THEN statement which combines the relational operators and the GOTO statement to make useful decisions.

In TI BASIC we have a more useful connection in addition to IF ... THEN) of IF ... THEN ... ELSE which allows us to make a conditional branch in a program. The computer evaluates the expression following the word IF in your program. If the expression is true the computer will jump to the line number which follows the word THEN . If the expression is false the computer will jump to the line following the word ELSE in your program.

Here is an example of what we could use this statement for. Supposing we are only interested in a sum of money above a certain figure, say £5, for our calculations in a program. We could write:

```
10   INPUT "ENTER AMOUNT IN POUNDS : ":VALUE

20   IF VALUE <=5 THEN 30 ELSE 50

30   PRINT "INVALID AMOUNT"

40   GOTO 10

50   PRINT "VALID AMOUNT - LET'S PROCEED"
```

This program segment looks for an amount and stores it in the variable VALUE. Line 20 looks at the number contained in the variable VALUE. If this is less than or equal to 5, then Line 30 is executed, and the program prints:

```
INVALID AMOUNT
```

If the number stored in VALUE is not less than or equal to 5, then the ELSE part of the statement is executed and the program jumps to Line 50, printing:

```
VALID AMOUNT - LET'S PROCEED
```

There are many uses if this statement, especially in games, and we are lucky to have it with our lovely TI.

Here is another little manipulator for you.

ON ... GOTO

Another statement that can be used when a decision is required is ON ...GOTO statement. This statement is a way of selecting a line number from a list of line numbers in response to specified conditions.

The conditions are specified in an argument that immediately follows the keyword ON. The argument is usually an expression that is calculated as normal, however an ON ... GOTO statement needs an integer. An integer is a whole number, so if the answer to the expression has a value between whole numbers, that is it has numbers after the decimal point, it is rounded to the nearest whole number.

The value of the integer that is calculated specifies the position of the line number to be used in a list of line numbers that follow the keyword GOTO.

A typical ON … GOTO statement would be structured as follows:

```
ON (expression) GOTO (list of line numbers)
```

Here is an example of an ON ... GOTO statement:

```
                        1  2  3  4  5  6  7
                       /  /  /  /  /  /  /
                      /  /  /  /  /  /  /
     10 ON 3+2 GOTO 30,40,50,60,70,80,90
```

The integer from the expression after the keyword ON is 5, so the program will GOTO the line number that is in fifth place in the list, that is, Line 70.

Here is an example of how an ON ... GOTO statement might be used:

```
10 ON 3+2 GOTO 20,40,60,80,100

20 PRINT "3+2 = 1"

30 GOTO 110

40 PRINT "3+2 = 2"

50 GOTO 110

60 PRINT "3+2 = 3"

70 GOTO 110

80 PRINT "3+2 = 4"

90 GOTO 110

100 PRINT "3+2 = 5"

110 STOP
```

Because the expression after the keyword ON gives the integer 5, the program will GOTO the fifth line number in the list, line 100, and print:

```
3+2 = 5
```

Bye for now,

Christina.

BASIC ASSEMBLY

Steve Peacock...

T H E B A S I C A S S E M B L E R #12 By Steve Peacock

MUSIC AND OTHER SOUNDS

Creating music with assembly language on the TI-99/4A, is slightly more complicated than when using BASIC. To create a sound a table must be located in VDP RAM. A good place for this table is address >1000. At CPU address >83CC you need to tell the computer where the sound data can be found in the VDP RAM. You then need to set the rightmost bit of byte >83FD to 1. The sound is then activated by placing the value of >01 to address >83CE. Once the sound has started the VDP interrupts must be enabled and disabled with the instructions of LIMI 0 and LIMI 2. These instruction should be put in a frequently used loop.

In creating sounds, there are four generators, three for music, and one for noise. Generators 1,2, and 3 are used to create music and generator 4, is used to create a noise. I will explain how to write a musical note using terms that I have made up to describe the values written into the program. These terms will help explain what you are doing and are not necessarily the correct technical terms. These terms are TOTAL BYTES (TB), GENERATOR #/FREQUENCY (GF), GENERATOR #/VOLUME (GV), and TIME DURATION (TD).

TB--> The total bytes consist of the number of byte that each tone or multi-tone/noise takes to describe it with the exception of the TD byte. A single tone will take 3 bytes, -->TB (not counted) GF (2) and GV (1) and TD (not counted). A two tone cord will take 6 bytes, -->TB (not counted) GF (2) GV (1) GF (2) GV (1) TD (not counted). A noise requires only 2 bytes, TB (not counted) GF (1) GV (1) TD (not counted).

GF--> The four generators are labeled 8, A, C, and E. After the generator is specified the frequency is given. This consist of three hex digits, for music and one hex digit for a noise. For example a tone of 220 hertz on generator #1 would be >8C1F. If the same tone is to be produced on generator #2 the code would be >AC1F. Same note, different generator. A noise of -3, would be written >E2.

GV--> Each generator must be given a volume. On the TI computer there are only sixteen volumes to choose from, (I bet you thought there were 31). Zero is the maximum and F is the minimum. Try a program in TI Basic that changes from volume 2 to 3, while doing a 'CALL SOUND', you will not hear a change in the volume. Please note that when GF #1 is used, the GV MUST be for the same generator. When you are through with all sounds, the volume must be turned off. To do this set the volume to the minimum, using the code 'F'.

TD--> The time that a sound will play is written as a two digit hex number. Remember this byte IS NOT counted in the TB at the start of the coding. This time can be any thing from 0 to approximately 4250 miliseconds (>00 to >FF).

A one tone sound would look like this:

```
 TB  GF  GV  TD
 |||    |||  ||  (The 00 is a byte of padding)
>0387,>3595,>FF00
```

A two tone sound would look like this:

```
 TB  GF  GV   GF   GVTD
 |||    |||  |  |  ||||
>0687,>3595,>A72A,>B5FF
```

GENERATOR #/FREQUENCY (GF)

#1->8 #2->A #3->C #4->E

FREQUENCY TABLE:

NOTE	DATA	FREQ	NOTE	DATA	FREQ	NOTE	DATA	FREQ	NOTE	DATA	FREQ	NOTE	DATA	FREQ
A1	93F	110	A2	C1F	220	A3	E0F	440	A4	F07	880	A5	004	1760
A#1	03C	117	A#2	01E	233	A#3	00F	466	A#4	807	932	A#5	C03	1865
B1	A38	123	B2	51C	247	B3	20E	494	B4	107	988	B5	903	1976
C1	735	131	C2	C1A	262	C3	60D	523	C4	B06	1047	C5	503	2093
C#1	732	139	C#2	419	277	C#3	A0C	554	C#4	506	1109	C#5	203	2217
D1	A2F	147	D2	D17	294	D3	E0B	587	D4	F05	1175	D5	003	2349
D#1	F2C	156	D#2	816	311	D#3	40B	622	D#4	A05	1245	D#5	D02	2489
E1	72A	165	E2	315	330	E3	A0A	659	E4	505	1319	E5	A02	2637
F1	128	175	F2	014	349	F3	00A	698	F4	005	1397	F5	802	2794
F#1	D25	185	F#2	E12	370	F#3	709	740	F#4	C04	1480	F#5	602	2960
G1	B23	196	G2	D11	392	G3	F08	784	G4	704	1568	G5	402	3136
G#1	B21	208	G#2	D10	415	G#3	708	831	G#4	304	1661	G#5	202	3322

GENERATOR #/VOLUME (GV)

Number
#1->9 #2->B #3->D #4->F
Volume
0 (Loudest) to F (Quietest)

TIME DURATION (TD)

M.SEC.	CODE	M.SEC.	CODE	M.SEC.	CODE	M.SEC.	CODE
0	00	250	0F	500	1E	750	2D
1000	3C	1250	4B	1500	5A	1750	69
2000	78	2250	87	2500	96	2750	A5
3000	B4	3250	C3	3500	D2	3750	E1
4000	E0	4250	FF				

The 8 noises are coded like this:

-1	-2	-3	-4	-5	-6	-7	-8
0	1	2	3	4	5	6	7

A -3 noise, with a volume of 5, for 1000 miliseconds would look like this:

```
 IBGF  GVTD
 ||||  ||||
>02E2,>F53C
```

A three note cord with a noise of -5:

```
>0B87,>3590,>A72A,>B0CB,>23D0,>E4F0,>FF00
```

Chicago TImes

```
0B===>TOTAL BYTES (11)
8735=>GENERATOR/FREQUENCY #1, 90=>GENERATOR/VOLUME #1
A72A=>GENERATOR/FREQUENCY #2, B0=>GENERATOR/VOLUME #2
CB23=>GENERATOR/FREQUENCY #3, D0=>GENERATOR/VOLUME #3
E4===>GENERATOR/FREQUENCY #4, F0=>GENERATOR/VOLUME #4
FF===>TIME DURATION
00===>PADDING
```

Padding will be added only if your coding needs it to make it have four digits at the end. If your music has multiple notes it should be strung together.

The above tables give you some of the frequencies and times that are available to you in creating music. If you need some frequencies, or times, that I have not listed, there is a program that will calculate them for you.
I have called the program AS/MUS/CAL, for ASsembly MUSic CALculator.

##

```
100 REM PROGRAM BA12B==>Basic Assembler #12 BASIC Version
110 REM MUSIC
120 REM (C)1986 S. PEACOCK
130 REM YOU MAY WANT A 'CALL CLEAR' HERE
140 CALL SCREEN(4)
150 DEF1$="AA55AA55AA55AA55"
160 DEF2$="3C3C3C3C3C3C3C3C"
170 DEF3$="00000000183C1810"
180 DEF4$="387CFEFEFE7C3810"
190 P1$="aaaaaaaa"
200 P2$="h"
210 P3$="p"
220 CALL MAGNIFY(2)
230 CALL CHAR(128,DEF4$)
240 CALL CHAR(129,DEF4$)
250 CALL CHAR(130,DEF4$)
260 CALL CHAR(131,DEF4$)
270 CALL SPRITE(#1,128,8,100,25,-5,-8,#2,129,5,100,50,-8,-3,#3,130,7,100,128,-8,
5,#4,131,16,100,200,-5,3)
280 CALL COLOR(9,16,15)
290 CALL COLOR(10,7,1)
300 CALL COLOR(11,16,1)
310 CALL CHAR(97,DEF1$)
320 DISPLAY AT(17,13):P1$
330 DISPLAY AT(18,13):P1$
340 DISPLAY AT(19,13):P1$
350 DISPLAY AT(20,13):P1$
360 CALL CHAR(104,DEF2$)
370 DISPLAY AT(13,17):P2$
380 DISPLAY AT(14,17):P2$
390 DISPLAY AT(15,17):P2$
400 DISPLAY AT(16,17):P2$
410 CALL CHAR(112,DEF3$)
420 DISPLAY AT(12,17):P3$
430 REM TO SEE HOW SOUND IS DONE ON THE TI THE CODING BELOW IS ALL THAT IS NEEDE
D.
440 REM THE ABOVE IS SOME GRAPHICS TO DRESS UP MY CELERBRATION OF ONE YEAR OF TH
E BASIC ASSEMBLER.
450 DIM T(29),F(29)
460 FOR L=1 TO 29
470 READ F(L),T(L)
```

```
480 NEXT L
490 FOR L=1 TO 29
500 CALL SOUND(T(L),F(L),0)
510 NEXT L
520 GOTO 490
530 DATA 262,250,44000,17
540 DATA 262,118,294,250
550 DATA 262,250,349,250
560 DATA 330,500,262,250
570 DATA 44000,17,262,118
580 DATA 294,250,262,250
590 DATA 392,250,349,500
600 DATA 262,250,44000,17
610 DATA 262,118,523,200
620 DATA 440,200,349,200
630 DATA 330,200,294,1000
640 DATA 466,250,44000,17
650 DATA 466,118,440,250
660 DATA 349,250,392,250
670 DATA 349,1000
680 END
```

```
################################################################################
********************************************************************************
*
*PROGRAM BA12A==>Basic Assembler #12 Assembly Version
*MUSIC
*(C)1986 S. PEACOCK
*
********************************************************************************
        REF   VMBW,VSBW,VWTR
        DEF   START
DEF1    DATA >AA55,>AA55,>AA55,>AA55  *CHAR 97d, 61h
DEF2    DATA >3C3C,>3C3C,>3C3C,>3C3C  *CHAR 104d, 68h
DEF3    DATA >0000,>0000,>183C,>1810  *CHAR 112d, 70h
DEF4    DATA >387C,>FEFE,>FE7C,>3810  *SPRITE DEF
ATT     DATA >6419,>8007,>6432,>8104  *SPRITE ATTRIBUTES
        DATA >6480,>8206,>64C8,>830F
        DATA >D000
MOT     DATA >FBF8,>0000,>F8FD,>0000  *SPRITE MOTION
        DATA >F805,>0000,>FB03,>0000
P1      TEXT 'aaaaaaaaa'              *TEXT TO BE
P2      TEXT 'h'                      *PRINTED
P3      TEXT 'p'                      *LATER
****************************PUT FOUR SPRITES ON SCREEN
START   LI    R5,>0400      *
        MOVB  R5,@>837A
        LI    R0,>01E1      *MAGNIFICATION 2
        BLWP  @VWTR
        LI    R0,>0400
CON     LI    R1,DEF4       *WRITE DEFINITION OF SPRITES
        LI    R2,8
        BLWP  @VMBW
        AI    R0,8
        CI    R0,>0418      *FOUR SPRITES TO BE WRITTEN
        JLE   CON
        LI    R0,>0300
        LI    R1,ATT        *WRITE ATTRIBUTES OF SPRITES
        LI    R2,17
        BLWP  @VMBW
        LI    R0,>0780
        LI    R1,MOT          *WRITE MOTION OF SPRITES
        LI    R2,16
        BLWP  @VMBW
***************************
        LI    R0,>038C        *WRITE COLOR OF CHAR SET #9
        LI    R1,>FE00        *WHITE ON GRAY
        BLWP  @VSBW
        LI    R0,>038D        *WRITE COLOR OF CHAR SET #10
        LI    R1,>6000        *DARK RED ON TRANSPARENT
        BLWP  @VSBW
        LI    R0,>038E        *WRITE COLOR OF CHAR SET #11
        LI    R1,>F000        *WHITE ON TRANSPARENT
        BLWP  @VSBW
*************************
        LI    R0,>0B08        *WRITE CHAR DEF TO CHAR 97d 61h
        LI    R1,DEF1
        LI    R2,8
        BLWP  @VMBW
        LI    R0,524          *PRINT NINE 'a's
PP1     LI    R1,P1           *
        LI    R2,9            *    (THE CAKE)
        BLWP  @VMBW           *
        AI    R0,32           *
        CI    R0,620          *PRINT FOUR ROWS OF THE 'a's
        JLE   PP1
*********************
        LI    R0,>0B40        *WRITE CHAR DEF OF CHAR 104d 68h
        LI    R1,DEF2
        LI    R2,8
        BLWP  @VMBW
        LI    R0,400          *PRINT ONE 'h'
PP2     LI    R1,P2           *
        LI    R2,1            *    (THE CANDLE)
        BLWP  @VMBW           *
        AI    R0,32           *
        CI    R0,496          *PRINT A COLUMN OF 4 'h'
        JLE   PP2
*********************
        LI    R0,>0B80        *WRITE CHAR DEF OF CHAR 112d 70h
        LI    R1,DEF3
        LI    R2,8
        BLWP  @VMBW
        LI    R0,368          *PRINT ONE 'p'
        LI    R1,P3           *    (THE FLAME)
        LI    R2,1
        BLWP  @VMBW
****************************************
*
*TO SEE HOW SOUND IS DONE ON THE TI THE
*CODING BELOW IS ALL THAT IS NEEDED.
*THE ABOVE IS SOME GRAPHICS TO DRESS UP
*MY CELEBRATION OF ONE YEAR OF THE
*BASIC ASSEMBLER. IF YOU CHOOSE TO NOT
*TYPE IN THE ABOVE DO NOT FORGET TO
*USE THE REF/DEF ITEMS THAT YOU NEED.
```

```
*
****************************************
SOUND  LI   R0,>1000       *SOUND LIST TO BE WRITTEN TO >0100
       LI   R1,SL          *SOUND LIST FOUND AT LABEL SL
       LI   R2,150         *IT IS 150 BYTES LONG
       BLWP @VMBW
       MOV  R0,@>83CC      *LOAD POSITION OF SOUND LIST INTO >83CC
       MOVB @CV,@>83CE     *LOAD >01 INTO >83CE, TO START SOUND
       SOCB @CV,@>83FD     *SET BIT 7 OF BYTE >83FD
       CLR  R7             *CLEAR REG. 7
LP     LIMI 2              *ENABLE VDP INTERRUPTS  !NEEDED FOR SOUND
       LIMI 0              *DISABLE VDP INTERRUPTS !AND SPRITES
       CB   R7,@>83CE      *IS >83CE ZERO, IS SOUND LIST FINISHED?
       JNE  LP             *IF NOT STAY IN LOOP
       JMP  SOUND          *IF IT IS PLAY SONG AGAIN
CV     DATA >0100          *DATA TO PREPARE SOUND GENERATION
SL     DATA >038C,>1A90,>0F03,>8300,>9F01  *
       DATA >038C,>1A90,>0703,>8D17,>900F  **
       DATA >038C,>1A90,>0F03,>8014,>900F  * *
       DATA >0383,>1590,>1E03,>8C1A,>900F  *  *
       DATA >0383,>009F,>0103,>8C1A,>9007  *   *
       DATA >038D,>1790,>0F03,>8C1A,>900F  *    *
       DATA >038D,>1190,>0F03,>8014,>901E  *     *
       DATA >038C,>1A90,>0F03,>8300,>9F01  *      *SOUND LIST, 150 BYTES
       DATA >038C,>1A90,>0703,>860D,>9012  *     *
       DATA >038E,>0F90,>1203,>8014,>9012  *    *
       DATA >0383,>1590,>1203,>8D17,>903C  *   *
       DATA >0380,>0F90,>0F03,>8300,>9F01  *  *
       DATA >0380,>0F90,>0703,>8E0F,>900F  * *
       DATA >0380,>1490,>0F03,>8D11,>900F  **
       DATA >0380,>1490,>3C03,>9FBF,>DF00  *
       END
```

A few people have asked how to detect the differences between different versions of XB and different peripheral cards when writing a program. The method I used in XBasher to detect MYARC XB II or TI XB (or one of its offshoots, i.e. Mechatronic, GK XB, etc) was:

```
10 CALL VERSION(X)
20 IF X=120 THEN RUN "Triton Super Extended BASIC version"
30 IF X>=300 THEN RUN "Geneve 9640 Advanced BASIC version"
40 IF X>=200 THEN RUN "MYARC Extended BASIC II version"
50 RUN "TI / GK Utility I / Mechatronics Extended BASIC version"
```

To detect between a CorComp and TI disk controller card, the method I used in my modification of REDISKIT was:

```
10 ON ERROR 40
20 DELETE "LD-CMDS"
30 RUN "CorComp version" 40 RUN "TI version"
```

I don't know an easy method (or even a hard method) to detect the MYARC controller. Does anyone out there have any bright ideas?

FORTH

ROB WILLIAMS.

Two useful FORTH words which are found in the FORTH-79 standard, but which are not part of fig-FORTH or TI-FORTH, are PICK and ROLL.

This is how they work:

Suppose that you have four numbers on the main Parameter Stack, or simply Stack, the numbers 1 2 3 4.

3 PICK will make a copy of the third number down on the stack, onto the top of the stack. This means that you will be left with 1 2 3 4 2 because 2 has been copied to the top of the stack.

Suppose the stack once more again holds 1 2 3 4

3 ROLL will put the third number down on the stack (ie the 2) on top of the stack, whilst pushing the numbers that were above it one place down, leaving 1 3 4 2.

Note that 2 PICK is the same as OVER, 2ROLL is the ame as SWAP, and 3 ROLL is the same as ROTATE. That is to say they do the same job.

I have defined the two words in FORTH assembler, which makes them run quite a lot faster than if they were defined in high level FORTH. In fact PICK is a factor of 3 faster than one high level PICK i've used, and ROLL is a factor of 15 faster than a high level ROLL found in the FORTH programming book!

To use them, type in the contents of SCREEN 264 and 265 which are listed below. It doesn't matter what screen number you use.

After they have been typed in with the FORTH editor and saved, you can access them by typing 264 LOAD and pressing <ENTER>. (Or whatever first number screen you have used).

The Hexadecimal coding in lines 7, 9, 10, 11 and 12 of SCREEN 264 is the part that is actually compiled by FORTH. The rest of the two SCREEN contents consists of comments, and the assembler mnemonics for PICK and ROLL. If you wanted to, you could re-assemble PICK and ROLL using these mnemonics. But having the HEX code referred to above means that you DO NOT have to have the FORTH assembler in memory before PICK and ROLL can be loaded and used.

If you want to re-assemble, you will need to remove the brackets from around the assembler portion of PICK on SCREEN 264, which I put in to turn the whole 5 lines into a comment.

I have seen these two FORTH words used extensively in at least one FORTH book, (FORTH PROGRAMMING by L.J. Scanlon) so I hope they come in useful to you.

Happy FORTH-ing!

=================

SCR #264

```
 0 ( "PICK" WRITTEN IN FORTH ASSEMBLER 18.9.85)
 1 ( CODE PICK  ) ( n1 -- n2 )
 2 (  *SP 0   MOV,   Move address index n1 into R0
 3       0 1   SLA,   Double it for word addressing
 4     SP 0     A,   Add stack to address to index
 5   0 *? *SP MOV,   Pick target no up,put on top of stack
 6           NEXT    Return     ) HEX
 7 CODE PICK C019 , 0A10 , A009 , C650 , 045F ,
 8
 9 CODE ROLL C019 , C099 , 0201 , 0001 , 8040 , 1602 , 05C9 ;
10           100F , 0A10 , C049 , A009 , C0D0 , 0602 , C100 ,
11           0644 , C414 , 0640 , 0644 , 0602 , 16FB , C241 ,
12           05C9 , C643 , 045F , DECIMAL
13
14
15                                      From TIT-BITS Vol 4 No 2 Nov 1985
                                        TI Users - Perth  (Australia)
```

SCR #265

```
 0 ( "ROLL" WRITTEN IN FORTH ASSEMBLER 19.9.1985 )
 1 CODE ROLL ( n -- )
 2  *SP 0  MOV, ( n into R0)      *SP  2 MOV, ( n also into R2 )
 3     1 1   LI, ( is n = 1 ? )     0  1   C,
 4       EQ  IF, ( if so ...  )      SP INCT, ( drop n )
 5         ELSE, ( if n <> 1 )      0  1 SLA, ( n*2 for word addr)
 6    SP 1  MOV, ( S ave SP R1 )    SP 0    A, ( top st adr+index )
 7 0 *? 3  MOV, ( target in R3 )      2 DEC, ( counter for loop )
 8     0 4  MOV, ( adr t in R4 )       4 DECT, ( no. to be moved )
 9       BEGIN, ( shift all nos) 4 *? 0 *? MOV, ( down one place)
10      0   DECT, ( next addr )        4 DECT, ( next adr for no's)
11     1 SP DEC, ( done ? )          EQ UNTIL, ( cont if R2<>0 )
12     1 SP MOV, ( restore SP )      SP INCT, ( adjust [n removd])
13   3 *SP MOV, ( put target st top)   ENDIF,
14        NEXT, ( return )
```

```
          DM1000 REVISION RECORD
          ----------------------

 MODIFIED BY RALPH ROMANS:

 VER 2.2 HAS FIXES TO VERSION II:
       - WHEN CATALOGING DRIVE 2 AND LIST TO LIST DEVICE IT
         WOULD SHOW DSK1 NOT DSK2
       - IMPROPER ERROR HANDLING IN  LIST DEVICE
       - DOING REDO OR BACK WHILE ENTERING LIST DEVICE OR
         LIST DEVICE CONTROL CODES CAUSED VARIOUS FAULTS.
       - A CALL KEY(3  IN LOADER PROGRAM CAUSED PROBLEMS
       - UNABLE TO CATALOG 127 FILES
       - LIST DEVICE WAS OPENED IN UPDATE MODE WHICH PREVENTED
         LIST CATALOG OF SEVERAL DISKS TO ONE DISK FILE
       - WHEN ASKING FOR FILE NAME ETC. IT WAS POSSIBLE TO
         MOVE CURSOR PASS END OF NAME
       - WHEN ENTERING CONTROL CODES ENTERING A SINGLE
         NUMBER CAUSED SYSTEM TO CRASH.
 2.2 ENHANCEMENTS:
       - ABLE TO DISPLAY ON SCREEN DISPLAY TYPE VAR 80
         OR FIXED 80 FILES
         FCTN-BACK OR  FCTN-REDO IS USED TO ABORT
         DISPLAYING FILE.
       - ABLE TO CHANGE SCREEN AND TEXT COLOR AND SAVE
         IT TO DISK WHEN LIST DEVICE IS SAVED.
       - FCTN-BACK WILL TAKE TO BACK TO THE PREVIOUS
         MENU UNTIL YOU REACH THE MAIN MENU.
       - LIST DEVICE IS OPENED IN DIS/VAR 80 APPEND MODE
         PERMITTING A DISK FILE TO CONTAIN CATALOG OF
         SEVERAL DISKS.
       - IN SECTOR COPY COPIES 103 SECTORS PER PASS
       - ADDED TI CLUB ADDRESS TO TITLE SCREEN
 VER 2.3 FIXES TO VER 2.2:
       - ABLE TO DISPLAY A FILE WHICH IS 10 CHARACTERS
         LONG
       - FCTN-BACK AND FCTN-REDO DIDNOT WORK PROPERLY
         IN A FEW AREAS
 2.4 ENHANCEMENTS:
       - THE FILE 'MGR1' IS OBTAINED FROM THE LAST DRIVE
         USED TO GET THE PRINTER CODES AND SCREEN COLORS
         INSTEAD OF ALWAYS DRIVE 1.
       - IN FILE OPTION 1, A RUNNING TOTAL OF FILE SIZE
         TO COPY/MOVE OR DELETE IS SHOWN
       - THE ABILITY TO LOAD AND RUN A PROGRAM IMAGE FILE HAS BEEN
         ADDED TO THE FILE UTILITIES.
 THE MAIN PURPOSE OF THIS LOADER IS TO LOAD AND RUN THOSE PROGRAM IMAGE FILES
 THAT RUN WITH 'GLOADER' OR 'GAMELOADER' IN XBASIC.

 THE FILE 'LOAD' SUPPLIED WITH DM1000 TO LOAD DM1000 FROM XBASIC HAS
 GLOADER EMBEDDED WITHIN THE XBASIC PROGRAM. THE FILE 'LOAD' WILL
 USE THE LAST DISK DRIVE USED TO LOAD ANY FILES(TI/CORCOM ONLY)

 ..XB VDP  MEANS THAT THE 9918 Video Display Processor REGISTERS ARE LEFT AS
           THEY ARE IN XBASIC. SOME PROGRAMS REQUIRE THIS TO RUN PROPERLY.
           THIS IS THE SAME AS POKING A NON ZERO IN THE XBASIC GLOADER PROGRAM.


 ..E/A VDP MEANS THAT THE 9918 VDP REGISTERS ARE THE SAME
           AS THEY ARE IN E/A. SOME PROGRAMS REQUIRE THIS TO RUN PROPERLY.
           THIS IS THE SAME AS POKING A ZERO IN THE XBASIC GLOADER PROGRAM.

 DM1000 DOES A 'CALL INIT' IF THE XBASIC MODULE IS INSTALLED SO IMAGE
 PROGRAMS THAT USE THE BUILT IN UTILITIES OF XBASIC WILL WORK.
```

```
NOTE :   A PROGRAM IMAGE FILE THAT IS LOADED WITH THE E/A MODULE OPTION 5
----     MAY NOT WORK WITH DM1000 LOADER IF THAT PROGRAM USES THE BUILT IN
         UTILITIES OF THE E/A.


VER 3.0 FIXES TO VER 2.4:
         - INCORRECT FILE COUNT WHEN GOING FROM 'M' TO 'C'
         - FILE COPY WOULD GIVE YOU A BAD COPY IF THE FILE BEING COPIED
           WAS STORED ON THE MASTER DISK AS A NON CONTINOUS FILE AND THE
           SIZE OF THE FIRST SEGMENT WAS EXACTLY 39 SECTORS WITH
           ADDITIONAL SECTERS IN ANOTHER SEGMENT ON THE DISK.
VER 3.1 FIXES TO VER 3.0:
         - FILE COPY WOULD GIVE YOU A BAD COPY IF THE MASTER FILE
           WAS A FRACTERED FILE OF EXACTLY 39 SECTORS AND THE SAME FILE
           NAME WAS ON THE COPY DISK.
         - WHEN ENTERING A FILE NAME IN VARIOUS MODES,
           IT WAS POSSIBLE TO MESS IT UP.
UNFIXED BUGS IN VER 3.1 - UNABLE TO DISPLAY SOME DIS/VAR 80 FILES THAT
                          ARE FULL OF CONTROL CHARACTERS. COMPUTER
                          HANGS UP!

VER 3.3-CHANGED DEFAULTS ON SWEEP AND DISK INITIALIZATION
         - DISK INITIALIZATION WORKS FOR MYARC AND CORCOM
         - READ/WRITE ERRORS GETS CLEARED AFTER 1ST USE ON DISK COPY
         - FILE 'MGR1' MAY NOW BE CALLED ANY NAME AND ALL FEATURES OF DM1000
           WILL WORK.!! THIS WILL ONLY WORK WITH TI CONTROLLER
           AND CORCOM CONTROLLER
         - THE LOADER FOR MYARC CONTROLLER IS CALLED LOADMY
         - DURING DISK INITIALIZATION MENU, YOU CAN USE THE UP ARROW TO
           GO BACK TO PREVIOUS QUESTION.

VER 3.4- ABLE TO DELETE/MOVE/COPY 1 SECTOR FILES
         - ADDED 'UP ARROW ACTIVE' NOTICE WHEN UP ARROW WILL
           TAKE YOU BACK TO PREVIOS QUESTION.

VER 3.5- ABLE TO TYPE/PRINT DISPLAY VAR 80/FIXED 80 FILES
         WHILE THE FILE LISTING IS ON THE SCREEN BY PRESSING
         A 'T' FOR TYPE(DISPLAY) FILE TO SCREEN OR
         'P' FOR PRINT TO LIST DEVICE WITH OPTIONAL CONTROL
         CODES SENT TO PRINTER FIRST.
         THE 'P' AND 'T' FOR PRINT OR TYPE ARE ONLY VALID
         IN THE LEFT MOST FIELD.
       - 'EOF' noticed added in lower left corner of screen

       - DISPLAY VAR 80/FIXED 80 MENU REMOVED

VER 3.7- NEW DSR FOR RAMDISKS
       - DISPLAY TYPE OF DISK CONTROLLER IN YOUR SYSTEM(TI CC MY ON MAIN SCREEN)
       - ABLE TO TYPE DIS/VAR FILES WITH NULLS. BEFORE COMPUTER WOULD HANG.
       - ABLE TO PUT IN 3 DIGIT NUMBERS FOR PRINTER CONTROL CODES.
       - ABLE TO SAVE PRINTER SETUP TO DISK DM1000 WAS LOADED FROM.
         INCLUDING RAMDISKS.
       - MGR1/MGR2 MAY BE RENAMED TO ANY NAME EXCEPT ON MYRAC DISK CONTROLLER
       - RELEASE DMG1/DMG2 FOR GENEVE 9640
         ALL FEATURES OF 3.7 MAY NOT WORK IF DMG1/DMG2 NAMES ARE CHANGED.
```

```
At Main Menu:
-------------
FCTN-3  = Enter list device
          and control codes
          (default=PIO)
          option to save codes
          and screen colors
          to disk with
          '(PROGRAM NAME)' I.E. MGR1
          name on it.
```

```
With Catalog on Screen:
-----------------------
FCTN-7  = Print catalog
          to list device
          (default=PIO)


With File List on Screen:
-------------------------
FCTN-6   = Proceed with
           given Commands
```

FROM
* TopIcs - LA 99ers * *
AUGUST 1987

99 FORTRAN from LGMA Products

A Review of First Impressions
by Ralph Landrum, HUG member

Courtesy San Antonio

I recently bought the LGMA 99 FORTRAN package that is advertised in the new TENEX catalog. So far I've studied the manual and compiled and linked the example programs that come with the package. It is well planned for the user. The manual is well written. It will be clear to anyone the least bit familiar with FORTRAN at any level. It is clearly meant for people who use the TI99 in XBASIC, but who want compiled versions of their programs. Assembly language programmers can also use internal TI99 subroutines and their own assembled code within the structure.

WHY FORTRAN?

FORTRAN has a conversational syntax like BASIC, and is therefore easier to use for me than A/L or C. In fact, the LGMA package is actually a combination of BASIC and FORTRAN II, being a subset of FORTRAN 77, rather than FORTRAN IV as advertised. I am familiar with (though not a trained programmer in) several forms of BASIC, FORTRAN II, and IV.

FORTRAN uses true subroutines, which I need in what I want to do with a computer. XBASIC uses true subroutines also.

FORTRAN is a compileable language. I want to be able to compile to machine language for speed. BASIC is compileable in some versions (for example IBM PC), but noone has brought out a good compiler, using true subroutines, for the TI99.

SO, FORTRAN could let me have a more familiar language, using true sub-routines, but compiled for operating speed.

99 Fortran is sold by TENEX

THE LGMA 99 FORTRAN Package

LGMA Products, Box 210, RD4, Apple-Butter Hill Road, Coopersburg, PA, 18036, is a company unknown to me. Alan L. Beard signs letters for them. Their 99 FORTRAN package was advertised in the latest TENEX catalog for $49.95. The package comprises two disks of ver. 2.1.3, and an excellent manual.

One disk has the boot (in E/A, M/M, BASIC, or TIW); the Full-screen Editor, Optimized Compiler, Linker, Debug, and example programs. The second disk has an excellent object module library with 78 functions and subroutines, including math functions (both single and double precision), and all the graphics and sound functions of TI BASIC. Included are: CHAR, CHARPA, COLOR, DELAY, DELETE, DELSPRITE, FILES, GCHAR, HCHAR, JOYST, KEY, MAGNI, MOTION,POSITI, SCREEN, SET32, SET40, SOUND, VCHAR, WAIT.

I find the manual to be VERY well written and organized. It explains things very simply for average programmers like me, but it also goes into detail for those excellent systems programs who will want to use internal subroutines of the TI99 roms, or want to add their own assembled routines to the library. Of course, you can write FORTRAN functions and subroutines, compile them, and add them to the library. Whoever did the manual must be an expert programmer AND user.

Your system requires 32K, at least one SSSD disk drive, and E/A, TIW, XBASIC, or MM.

Remember that this FORTRAN is a SUBSET of FORTRAN 77, with a few extra features. For example, it does not support the ENTRY statement of FORTRAN 77, but it does support the DOWHILE statement form PASCAL-- NOT FORTRAN 77. It is a subset in other ways, of course, being shoehorned into a small computer. Its program limit it 2 segments of 8K each. Integer constants take 2 Bytes as do logical constants. Single-precision constants occupy 4 Bytes, while Double-precision ones occupy 8 Bytes. The author includes a section of the manual explaining various tricks of the system to save space.

IS THE PROGRAM WORTH THE MONEY?

If you are comparing the too cheap cost of the programs from Clint Pulley, and the FREE and from the heart contributions of Warran Agee, Ron Albright, and many others who gave and taught us our c99 language, then you will look at $50 as a lot.However, because of the quality of work, the completeness, and comparison with the cost of other commercial programs, I find it reasonable.

I have not tried to program and run benchmarks against other programs, nor have I yet tested the optimizer by comparing routines like double- nested DO LOOPS compiled from source and written in assembler, but my elation in finding the system to be 77 instead of IV, the first programs I've compiled, the obvious effort of the author to make the system comparable to the XBASIC system we know with graphics and sound, and the excellent manual make me vote overwhelmingly YES, the program is more than I expected, and worth the money.

In the little time I've had to exercise the program, I find only two things I hope can be improved in future. One is to have a scale on the screen to tell me where I am on the eighty-column line. The second is to allow the LINKER program to automatically scan through more than one library disk just as it automatically iterates to let you load more than one OBJECT file. Those are not big objections ... they could just be made more convenient.

Extracted from CIN-DAY NEWS
June 1987

DISK DRIVE WOES

By, R.M. Bies

BYTE LINE Decatur 99er U.G. MARCH, 1987

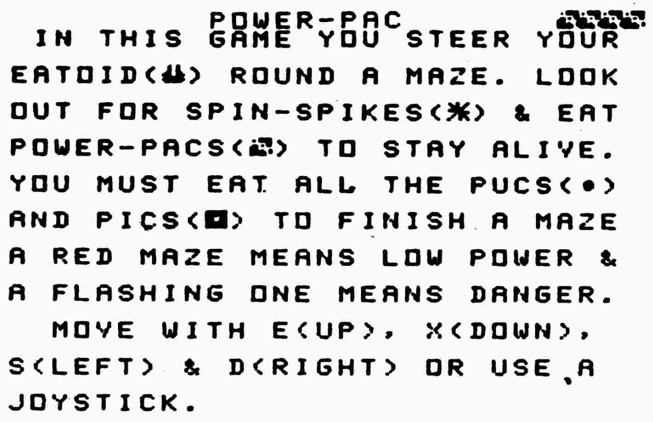
As our disk drives age, they become subject to problems which someone with the proper equipment can correct. (You have an indication of this problem when the drive will read and write reliably on a disk recently formatted on it, but the data so written cannot be read in another drive.) Also, with many of the half-height double sided drives, the heads are fragile, and if caught on the disk envelope, can be pulled out of position. Generally, the cost of replacement heads and the labor of replacement and alignment is greater than the cost of a new drive.

There are also less obvious sources of trouble not so exotic to remedy. I will deal with three which I have encountered: insecure mounting of the 12V regulator on the power-supply board, faulty power connections, and gummy head rails.

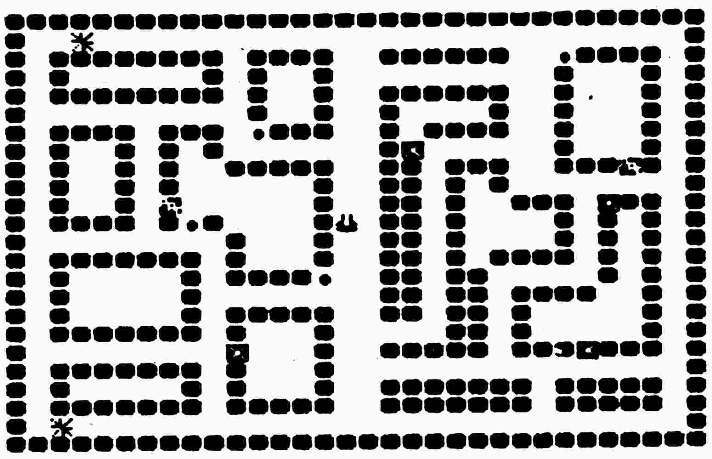
If it looks like the drive or drives are bogging down (paticularly in twin half-height instalations), and the drives ultimately stop and crash, it may be worthwhile to check the mounting of the 12V regulator. This is a TO-3 on the power-supply board, held down with screws. The board is its heat sink. I have found that tightening a loose mounting screw can alleviate this problem. (Tighten with care, of course.) You know the power-supply is being overloaded when the measured voltage on the 12V line on the drive drops off well below 12V as the drive slows.

If that wasn't it, and particularly when moving the drives around sometimes seems to fix it for a while, check the connections in the four power lines to the board on the disk drive, again, particularly in a twin drive installation. These connections come in two slightly different sizes and for some reason. the male pins seem to always be of the smaller size, the female of the larger size. It may help to spread the male pins slightly, or to close the female connectors slightly. They are not gold-plated so are subject to oxidation--contact cleaner may also help.

Now for the most obscure. One of my drives regularly would not read certain tracks on a disk, but the disk worked fine in the other drive. That the drive woud read most tracks suggested that the electronics was OK, that the other drive would read the whole disk suggested that it was not an ordinary alignment problem. It looked like a mechanical problem. Nothing seemed out of place with the drive removed, no foreign objects present. Yet, the heads seemed to offer slightly more resistance toward the end of their travel. A drop or two of lighter fluid on each rail with the assembly exercized by hand back and forth seem to yield a smoother action. Indeed it worked. The rails were gummy enough at the extreme of travel to defeat the electronic positioning procedure.



```
        POWER-PAC          ▦▦▦▦
 IN THIS GAME YOU STEER YOUR
EATOID(♟) ROUND A MAZE. LOOK
OUT FOR SPIN-SPIKES(✱) & EAT
POWER-PACS(▦) TO STAY ALIVE.
YOU MUST EAT ALL THE PUCS(●)
AND PICS(■) TO FINISH A MAZE
A RED MAZE MEANS LOW POWER &
A FLASHING ONE MEANS DANGER.
  MOVE WITH E(UP), X(DOWN),
S(LEFT) & D(RIGHT) OR USE A
JOYSTICK.
```



```
100 REM *********************
110 REM   P O W E R - P A C
120 REM     by Toby Grays
130 REM  TI.S.H.U.G.Under'18
140 REM  YOUNGER SET MEMBER
150 REM  (requires Ex/Basic)
160 REM       October'85
170 REM *********************
180 CALL CLEAR :: CALL SCREE
N(12):: RANDOMIZE :: DIM SX(
19),SY(19),R$(5)
190 FOR I=1 TO 5 :: READ R$(
I):: NEXT I
200 DATA "MAZE-NOVICE","MAZE
-PLAYER","MAZE-HERO","MAZE-S
TAR","MAZE-MASTER"
210 FOR I=1 TO 8 :: READ C,A
$ :: CALL CHAR(C,A$):: NEXT
I
220 DATA 138,0000183C3C3C18,
96,2424246GE77EFF7E1818183C7
E7EFF7E0A1FFF0707FF1F0A020F1
FFFFF1F0F02
230 DATA 100,7EFF7EE76624242
47EFF7E7E3C18181850F8FFE0E0F
FF85040E0F8FFFFF8E04
240 DATA 136,915234F81F2C4A8
9,137,894A2C1FF8345291,130,1
F5F036BE0E9FDFC,129,FFBDDBE7
E7DBBDFF,110,7CFEFEFEFEFE7C
250 CALL COLOR(9,7,1,10,5,1,
11,14,1,12,13,1,13,16,7)
260 A$=RPT$(" ",11):: A$=A$&
"POWER-PAC"&A$ :: FOR I=2 TO
 28
270 CALL HCHAR(1,I-1,ASC(SEG
$(A$,I-1,1))):: CALL HCHAR(1
,1,130,4):: NEXT I
280 RESTORE 300 :: FOR I=2 T
O 22 STEP 2 :: READ A$
290 DISPLAY AT(I,1):A$ :: NE
XT I :: DISPLAY AT(24,3):"PR
ESS ANY KEY TO PLAY."
300 DATA " IN THIS GAME YOU
STEER YOUR",EATOID(`) ROUND
A MAZE. LOOK,OUT FOR SPIN-SP
IKES0 & EAT
310 DATA POWER-PACS() TO ST
AY ALIVE.,YOU MUST EAT ALL T
HE PUCS(
          ),AND PICS() TO FI
NISH A MAZE
```

```
320 DATA A RED MAZE MEANS LO
W POWER &,A FLASHING ONE MEA
NS DANGER.," MOVE WITH E(UP
), X(DOWN),","S(LEFT) & D(RI
GHT) OR USE A",JOYSTICK.
330 CALL KEY(0,K,S):: IF S=0
 THEN 330
340 CALL CLEAR :: RESTORE 36
0 :: FOR I=1 TO 65 :: READ W
,X,Y,Z :: IF W=Y THEN 360
350 CALL HCHAR(X,W,110,Y-W+1
):: GOTO 370
360 CALL VCHAR(X,W,110,Z-X+1
)
370 NEXT I :: GOTO 480
380 DATA 1,1,32,1,32,1,32,24
,1,24,32,24,1,1,1,24,3,3,10,
3,12,3,15,3,18,3,23,3,26,3,3
0,3
390 DATA 30,4,30,9,30,11,30,
19,30,21,30,22,26,22,29,22,1
8,22,24,22,11,22,14,22,3,22,
9,22
400 DATA 3,20,3,21,3,14,3,18
,3,7,3,12,3,4,3,5,15,3,15,7,
15,9,15,15,15,17,15,22,18,5,
18,17
410 DATA 10,4,10,5,4,5,9,5,1
2,4,12,7,13,7,15,7,4,7,6,7,6
,8,6,12,4,12,5,12
420 DATA 8,7,10,7,10,8,10,8,
11,9,14,9,11,15,14,15,11,13,
11,14,8,12,10,12,8,8,8,11
430 DATA 4,14,9,14,9,15,9,18
,4,18,8,18,4,20,9,20,9,21,9,
22,11,17,15,17,11,18,11,21
440 DATA 19,5,23,5,23,6,23,7
,20,7,22,7,26,4,26,9,27,9,30
,9,19,8,19,17
450 DATA 21,9,21,18,22,9,23,
9,23,10,23,10,24,11,26,11,26
,12,26,14,23,14,25,14
460 DATA 28,11,30,11,24,19,2
9,19,24,16,24,18,25,16,27,16
,28,12,28,15
470 DATA 18,19,22,19,22,15,2
2,18,18,21,24,21,26,21,29,21
480 SR=0 :: LL=1
490 EN=80-LL*5 :: NS=LL*2-1
500 FOR I=0 TO NS
510 SX(I)=INT(1.5*I+3):: SY(
I)=2-21*(I/2=INT(I/2))
520 CALL HCHAR(SY(I),SX(I),1
36):: CALL SOUND(10,110,0)

530 NEXT I
540 YY=12 :: YX=16 :: YC=32
:: YP=96 :: N=(LL+7)/2 :: FO
R I=1 TO N*2
550 X=INT(30*RND)+2 :: Y=INT
(22*RND)+2 :: CALL GCHAR(Y,X
,P)
560 IF P<>110 THEN 550 :: CA
LL HCHAR(Y,X,129-9*(I>N))
570 NEXT I :: FOR I=1 TO N/2
580 X=INT(30*RND)+2 :: Y=INT
(22*RND)+2 :: CALL GCHAR(Y,X
,P):: IF P<>110 THEN 580
590 CALL HCHAR(Y,X,130):: CA
LL SOUND(100,1000,0):: NEXT
I
600 CALL HCHAR(YY,YX,YP)
610 FOR I=0 TO NS :: GOSUB 7
20
620 X=SX(I):: Y=SY(I):: P=13
7+(EN/2=INT(EN/2))
630 X1=SGN(YX-X):: Y1=SGN(YY
-Y)
640 CALL GCHAR(Y+Y1,X+X1,P1)
:: CALL GCHAR(Y+Y1,X,P2):: C
ALL GCHAR(Y,X+X1,P3)
650 IF (P1=32)+((P1>95)*(P1<
104))THEN 690
660 IF (P2=32)+((P2>95)*(P2<
104))THEN X1=0 :: GOTO 690
670 IF (P3=32)+((P3>95)*(P3<
104))THEN Y1=0 :: GOTO 690
680 CALL HCHAR(Y,X,P):: X1=0
 :: Y1=0 :: GOTO 710
690 CALL GCHAR(Y+Y1,X+X1,P1)
:: CALL HCHAR(Y,X,32):: CALL
 HCHAR(Y+Y1,X+X1,P)
700 IF P1<>32 THEN 960
710 SY(I)=Y+Y1 :: SX(I)=X+X1
 :: NEXT I :: GOTO 610
720 EN=EN-.5 :: IF EN<26 THE
N CALL COLOR(10,7,1):: IF EN
<11 THEN CALL COLOR(10,16,1)
730 CALL KEY(3,K,S):: IF (K<
>69)*(K<>68)*(K<>83)*(K<>88)
THEN 790
740 X=0 :: Y=0 :: IF K=69 TH
EN Y=-1
750 IF K=68 THEN X=1
760 IF K=83 THEN X=-1
770 IF K=88 THEN Y=1
780 GOTO 830
790 X=0 :: Y=0 :: CALL JOYST
(1,W,Z):: IF W=0 AND Z=0 THE
N CALL JOYST(2,W,Z)
800 IF W<>0 THEN X=W/4 :: GO
TO 830
810 IF Z<>0 THEN Y=-(Z/4)::
GOTO 830
820 GOTO 880
830 CALL GCHAR(YY+Y,YX+X,P):
: IF (P=110)THEN 880
840 IF X=0 THEN C=99+Y*2 ELS
E C=101+X*2
850 CALL HCHAR(YY,YX,32):: C
ALL HCHAR(YY+Y,YX+X,C+(EN=IN
T(EN)))
860 IF (P<>32)*((P<136)+(P>1
37))THEN 900
870 YY=YY+Y :: YX=YX+X :: IF
 (P>135)*(P<138)THEN 960
880 IF EN=0 THEN 950
890 RETURN
900 CALL HCHAR(YY+Y,YX+X,110
):: CALL HCHAR(YY,YX,96):: X
=0 :: Y=0
910 CALL SOUND(100,1000,10,-
1,0)
```

```
920 IF P=130 THEN EN=EN+INT(
LL+20*RND)+1 :: SR=SR+20 ::
CALL COLOR(10,5,1):: P=3
2 :: GOTO 870
930 SR=SR+(INT(LL*RND)+1)*10
 :: PE=PE+1 :: IF PE=N*2 THE
N 1030
940 P=32 :: GOTO 870

950 DISPLAY AT(12,8):"NO POW
ER!!" :: FOR D=1 TO 200 :: N
EXT D
960 FOR I=1 TO 30 :: CALL SO
UND(-100,-5,I):: NEXT I
970 CALL CLEAR :: PE=0 :: CA
LL COLOR(10,5,1)
980 DISPLAY AT(3,3):"YOU SCO
RED";SR;"POINTS.": :"   YOU
GOT UP TO LEVEL";LL
990 DISPLAY AT(8,8):"RATING:
":RPT$(" ",14-LEN(R$(LL/2))/
2);R$(LL/2)
1000 IF LL=10 THEN DISPLAY A
T(10,12) SUPREME"
1010 FOR D=1 TO 200 :: NEXT
D :: DISPLAY AT(24,1):"PRESS
 ANY KEY TO PLAY AGAIN."
1020 GOTO 330
1030 FOR I=2 TO 16 :: CALL S
OUND(-200,I*60,0):: CALL COL
OR(10,I,1,9,I-1,1):: NEXT I
1040 SR=SR+INT(EN)*10 :: IF
LL=10 THEN LL=9
1050 CALL COLOR(10,5,1,9,7,1
):: LL=LL+1 :: PE=0 :: CALL
HCHAR(YY,YX,32):: FOR I=0 TO
 NS
1060 CALL HCHAR(SY(I),SX(I),
32):: SX(I)=0 :: SY(I)=0 ::
SC(I)=0 :: NEXT I :: GOTO 49
0
```

```
100 ! *************
110 ! *BRIGHT EYES*
120 ! *************
130 ! TISHUG LIBRARY 110
140 ! EXTENDED BASIC
150 ! FROM WATERSHIP DOWN
160 ! BY MIKE BATT
170 ! CONVERTED BY R DUNLOP
180 CALL CLEAR
190 AF0=779 :: A0=825 :: AS0
,BF0=875 :: C0=982 :: CS0,DF
0=1040 :: D0=1102 :: DS0,EF0
=1168 :: E0=1237 :: F0=1311
:: FS0,GF0=1389
200 CALL CLEAR :: A1=110 ::
AS1,BF1=117 :: B1=123 :: C1=
131 :: CS1,DF1=139 :: D1=147
 :: DS1,EF1=156 :: E1=165 ::
 F1=175 :: FS1,GF1=185 :: G1
=196
210 A2=220 :: AS2,BF2=233 ::
 B2=247 :: C2=262 :: CS2,DF2
=277 :: D2=294 :: DS2,EF2=31
1 :: E2=330 :: F2=349 :: FS2
,GF2=370 :: G2=392
220 A3=440 :: AS3,BF3=466 ::
 B3=494 :: C3=523 :: CS3,DF3
=554 :: D3=587 :: DS3,EF3=62
2 :: E3=659 :: F3=698 :: FS3
,GF3=740 :: G3=784
230 A4=880 :: AS4,BF4=932 ::
 B4=988 :: C4=1047 :: CS4,DF
4=1109 :: D4=1175 :: DS4,EF4
=1245 :: E4=1319 :: F4=1397
:: FS4,GF4=1480 :: G4=1568
240 G0=1471 :: GS0,AF1=1559
:: GS1,AF2=208 :: GS2,AF3=41
5 :: GS3,AF4=831 :: GS4,AF5=
1661 :: R=40000
250 C=450 :: M=C*2 :: DC=C*1
.5 :: Q=C*0.5 :: COUNT=0 ::
FL=0 :: CALL MAGNIFY(4):: CA
LL PICTURE :: GOTO 360
260 CALL SOUND(T,S1,V1):: RE
TURN !1 NOTE
270 CALL SOUND(T,S2,V2):: RE
TURN !1 NOTE(S2)
280 CALL SOUND(T,S3,V3):: RE
TURN !1 NOTE(S3)
290 CALL SOUND(T,S1,V1,S2,V2
):: RETURN !2 NOTES(S1,S2)
300 CALL SOUND(T,S1,V1,S3,V3
):: RETURN !2 NOTES(S1,S3)
310 CALL SOUND(T,S2,V2,S3,V3
):: RETURN !2 NOTES(S2,S3)
320 CALL SOUND(T,S1,V1,S2,V2
,S3,V3):: RETURN !3 NOTES
330 CALL SOUND(T,S1,V1,S2,V2
,S3,30,-4,V3):: RETURN !BASS
 +2 NOTES
340 CALL SOUND(T,S3,30,S3,30
,S3,30,-4,V3):: RETURN !BASS
```

```
350 CALL SOUND(T,S1,V1,S3,30
,S3,30,-4,V3):: RETURN !BASS
 +1 NOTE
360 V1=10 :: V2=10 :: V3=6 :
:. T=Q :: S3=G0 :: GOSUB 340
:: S1=D2 :: GOSUB 350 :: S1=
G2 :: GOSUB 350 :: S1=D2 ::
GOSUB 350 :: S1=B3 :: GOSUB
350
370 S1=G2 :: GOSUB 350 :: S1
=D2 :: GOSUB 350 :: S1=G2 ::
 GOSUB 350 :: GOSUB 680 :: V
1=0 :: V2=6 :: V3=6
380 FOR COUNT=1 TO 2 :: V1=1
 :: T=Q :: S1=B3 :: S3=G0 ::
 GOSUB 350 :: S1=D3 :: S2=B3
 :: T=C :: GOSUB 330 :: T=Q
:: GOSUB 350 :: T=C :: GOSUB
 330
390 S1=B3 :: T=Q :: GOSUB 35
0 :: S1=D3 :: GOSUB 350 :: T
=M :: S1=E3 :: S2=C1 :: GOSU
B 290 :: S1=D3 :: S2=G0 :: G
OSUB 350
400 S3=G0 :: T=2*M :: GOSUB
340
410 S1=G2 :: S2=B2 :: S3=E0
:: T=C :: GOSUB 330 :: T=Q :
: S1=B3 :: GOSUB 260 :: S2=G
2 :: T=C :: GOSUB 330 :: S1=
G2 :: S2=B2 :: GOSUB 330
420 S1=B3 :: T=Q :: GOSUB 26
0 :: T=M :: S1=C3 :: S2=C1 :
: GOSUB 290 :: S1=B3 :: S3=G
0 :: GOSUB 350 :: T=1.5*M ::
 GOSUB 340 :: S3=G1 :: T=C :
: GOSUB 280
430 !@P-
440 T=Q :: S1=A3 :: S2=FS2 :
: S3=D1 :: GOSUB 320 :: GOSU
B 320 :: GOSUB 320 :: GOSUB
320 :: S3=C1 :: GOSUB 320 ::
 S1=B3 :: GOSUB 320
450 S1=C3 :: S2=A3 :: T=C ::
 GOSUB 320 :: S1=D3 :: S2=G2
 :: S3=G0 :: GOSUB 330 :: S2
=B3 :: S3=A1 :: GOSUB 320 ::
 T=M+DC :: S1=G2 :: S2=C1 ::
 GOSUB 290
460 T=Q :: S1=E2 :: S2=C1 ::
 GOSUB 290 :: S1=C3 :: S2=E2
 :: S3=A1 :: GOSUB 320 :: T=
C :: GOSUB 320 :: T=Q :: GOS
UB 320
470 S1=FS2 :: S3=D0 :: T=M*2
 :: GOSUB 350 :: S3=D1 :: T=
M+Q :: GOSUB 280 :: S1=G2 ::
 S2=E2 :: T=C :: GOSUB 320
480 T=Q :: S1=A3 :: S2=FS2 :
: GOSUB 320 :: T=C :: S1=B3
:: S2=G2 :: S3=G0 :: GOSUB 3
30 :: T=Q :: S1=D3 :: GOSUB
350 :: T=C :: S2=B3 :: GOSUB
 330
```

```
490 T=Q :: S1=B3 :: GOSUB 35
0 :: T=C :: S1=D3 :: S2=B3 :
: GOSUB 330 :: T=M :: S1=E3
:: S2=C1 :: GOSUB 290 :: S1=
D3 :: S3=G0 :: GOSUB 350
500 T=M+C+Q :: GOSUB 340 ::
S1=G2 :: T=Q :: GOSUB 350 ::
 T=M :: S3=E0 :: GOSUB 350 :
: T=Q :: S1=B3 :: S2=B2 :: G
OSUB 330
510 T=C :: S1=G2 :: GOSUB 33
0 :: T=Q :: S1=B3 :: S2=G2 :
: GOSUB 330 :: T=M :: S1=C3
:: S2=C1 :: GOSUB 290 :: S1=
B3 :: S3=G0 :: GOSUB 350
520 T=M+C+Q :: GOSUB 340 ::
T=Q :: S1=G2 :: GOSUB 350 ::
 S1=A3 :: S2=FS2 :: S3=D1 ::
 GOSUB 320 :: GOSUB 320 :: G
OSUB 320 :: T=C :: GOSUB 320
530 T=Q :: S3=C1 :: GOSUB 32
0 :: S1=B3 :: GOSUB 320 :: T
=C :: S1=C3 :: S2=A3 :: GOSU
B 320 :: S1=D3 :: S2=G2 :: S
3=G0 :: GOSUB 330
540 S2=B3 :: S3=A1 :: GOSUB
320 :: T=M+DC :: S1=G2 :: S2
=C1 :: GOSUB 290 :: T=Q :: G
OSUB 290 :: S1=C3 :: S2=E2 :
: GOSUB 320
550 T=C :: GOSUB 320 :: T=Q
:: GOSUB 320 :: T=2*M :: S1=
B3 :: S2=DS1 :: GOSUB 290 ::
 T=M :: S1=AS3 :: S2=CS1 ::
GOSUB 290
560 S1=B3 :: S2=D1 :: GOSUB
290 :: T=Q :: S1=C3 :: S2=FS
2 :: S3=D0 :: GOSUB 330 :: T
=C :: GOSUB 330 :: S1=FS2 ::
 T=Q :: GOSUB 350
570 T=M*2 :: S1=G2 :: S3=G0
:: GOSUB 350 :: T=M :: S3=G0
 :: GOSUB 340 :: T=Q :: S1=B
3 :: S2=G2 :: GOSUB 330
580 S1=D3 :: T=C :: GOSUB 33
0 :: T=2*M+Q :: S3=B1 :: GOS
UB 300 :: T=DC :: S1=G2 :: S
2=C1 :: GOSUB 290 :: S1=C3 :
: S2=G2 :: S3=C1 :: GOSUB 32
0
590 T=Q :: S1=B3 :: S2=E2 ::
 GOSUB 320 :: S1=A3 :: GOSUB
 320 :: S1=B3 :: S2=FS2 :: S
3=D1 :: GOSUB 320 :: S1=A3 :
: T=2*M+DC :: GOSUB 320
600 S1=B3 :: S2=FS2 :: S3=D0
 :: T=Q :: GOSUB 330 :: S1=D
3 :: T=C :: GOSUB 330 :: T=2
*M+Q :: S3=G0 :: GOSUB 350 :
: S1=G2 :: S2=E2 :: S3=C1 ::
 T=Q
610 GOSUB 320 :: GOSUB 320 :
: GOSUB 320 :: T=DC :: S1=C3
 :: S2=G2 :: GOSUB 320 :: T=
Q :: S1=B3 :: GOSUB 300 :: S
1=A3 :: T=2*M+Q :: S3=A1 ::
GOSUB 300
620 T=2*M :: GOSUB 280 :: T=
Q :: S1=B3 :: S2=FS2 :: S3=D
S0 :: GOSUB 330 :: GOSUB 330
 :: GOSUB 330 :: T=DC :: S3=
E0 :: GOSUB 330
630 T=Q :: S1=G2 :: GOSUB 35
0 :: S1=D3 :: S2=A3 :: S3=FS
0 :: T=DC :: GOSUB 330 :: T=
Q :: GOSUB 330 :: T=DC :: GO
SUB 330
640 S1=G2 :: S2=D2 :: S3=G0
:: GOSUB 330 :: T=Q :: S1=C3
 :: S2=G2 :: S3=C0 :: GOSUB
330 :: GOSUB 330 :: GOSUB 33
0
650 T=DC :: S1=D3 :: S2=E2 :
: S3=C1 :: GOSUB 320 :: S1=E
3 :: T=C :: S2=G2 :: GOSUB 3
20 :: T=DC :: S1=A3 :: S2=E2
 :: S3=A1 :: GOSUB 320
660 IF COUNT<>1 THEN 790
670 T=Q :: S1=C3 :: S2=FS2 :
: S3=D0 :: GOSUB 330 :: T=C
:: S1=B3 :: GOSUB 330 :: T=M
*2+Q :: S1=G2 :: S3=G0 :: GO
SUB 350
680 V1=10 :: V2=10 :: V3=6 :
: T=Q :: S1=D3 :: S2=B3 :: G
OSUB 330 :: S2=G2 :: GOSUB 3
30 :: S2=D2 :: GOSUB 330 ::
S2=G2 :: GOSUB 330
690 S2=B3 :: GOSUB 330 :: S1
=E3 :: GOSUB 330 :: S1=FS3 :
: GOSUB 330 :: S1=G3 :: GOSU
B 330 :: T=C :: S1=A4 :: S3=
E0 :: GOSUB 330
700 T=Q :: S1=G3 :: GOSUB 33
0 :: T=C :: S1=B3 :: S2=G2 :
: GOSUB 330 :: S2=E2 :: T=Q
:: GOSUB 330 :: S1=G2 :: GOS
UB 330 :: S1=E2 :: GOSUB 330
710 S2=G2 :: GOSUB 330 :: S2
=E2 :: GOSUB 330 :: S2=B2 ::
 GOSUB 330 :: S2=E2 :: GOSUB
 330 :: S1=G2 :: S2=E2 :: GO
SUB 330 :: S1=A3 :: GOSUB 33
0
720 S1=G2 :: GOSUB 330 :: S1
=A3 :: GOSUB 330 :: S1=G2 ::
 S2=R :: S3=C1 :: GOSUB 320
:: S2=C2 :: GOSUB 320 :: S2=
E2 :: GOSUB 320
730 S2=C2 :: GOSUB 320 :: S2
=E2 :: GOSUB 320 :: S2=C2 ::
 GOSUB 320 :: S2=G1 :: GOSUB
 320 :: S1=C2 :: GOSUB 320 :
: S2=E2 :: GOSUB 320
740 S2=C2 :: GOSUB 320 :: S2
=E2 :: GOSUB 320 :: S2=C2 ::
 GOSUB 320 :: S2=E2 :: GOSUB
 320 :: S1=A3 :: GOSUB 320 :
: S1=G2 :: GOSUB 320
750 S1=A3 :: GOSUB 320 :: V1
=0 :: V2=6 :: V3=6 :: IF COU
NT=0 THEN RETURN
760 !@P+
770 NEXT COUNT
780 !@P-
```

→

```
790 T=Q :: S1=C3 :: S2=FS2 :
: S3=D0 :: GOSUB 330 :: T=C
:: S1=B3 :: GOSUB 330 :: T=M
*2+Q :: S1=G2 :: S3=G0 :: GO
SUB 350
800 !@P+
810 IF FL=1 THEN 850
820 FL=1
830 !@P-
840 T=M :: GOSUB 340 :: T=Q
:: S1=B3 :: S2=G2 :: GOSUB 3
30 :: GOTO 580
850 !CODA
860 T=C :: GOSUB 340 :: T=Q
:: V1=8 :: V2=10 :: V3=7 ::
S1=B2 :: GOSUB 350 :: S1=D2
:: GOSUB 350 :: S1=B2 :: GOS
UB 350 :: S1=D2 :: GOSUB 350
870 S1=G1 :: GOSUB 350 :: S1
=B2 :: GOSUB 350 :: T=C :: G
OSUB 340 :: T=Q*1.1 :: GOSUB
 350 :: T=T*1.1 :: S1=D2 ::
GOSUB 350
880 T=T*1.1 :: S1=G1 :: GOSU
B 350 :: T=T*1.1 :: S1=B2 ::
 GOSUB 350 :: T=T*1.1 :: S1=
D2 :: GOSUB 350 :: T=T*1.1 :
: S1=G2 :: GOSUB 350
890 V1=4 :: V2=4 :: V3=4 ::
T=2*M :: S1=B3 :: S2=D2 :: S
3=G0 :: GOSUB 330
900 !@P+
910 FOR DEL=1 TO 1500 :: NEX
T DEL
920 CALL CLEAR :: CALL DELSP
RITE(ALL)
930 END
940 SUB PICTURE
950 CALL SCREEN(7):: DISPLAY
 AT(2,10):"BRIGHT EYES" :: D
ISPLAY AT(21,13):"FROM" :: D
ISPLAY AT(22,8):"WATERSHIP D
OWN"
960 CALL CHAR(96,"0000000107
0F3F7F",97,"FFFFFFFFFFFFFFFF
",98,"000000E0F0F8F8AC",99,"
ACFE8EFEFEFEFCFC")
970 CALL CHAR(100,"FFFFFFFFF
FFFFFFF",101,"FFFFFFFFFFFFFF
FF",102,"FCFCFCF8F8F8F0F0",1
03,"F0E0E0E0E0C0C0C0")
980 CALL CHAR(104,"FFFFFFFFF
FFFFFFE",105,"FEFEFCFCFCF8F8
F8",106,"C0C0808000000000",1
07,"0000000000000000")
990 CALL CHAR(108,"000000000
0000000",109,"0000031F7FFFFF
FF",110,"03060E3F7F7F7F7F",1
11,"7FFFFFFFFFFFFFFF")
1000 CALL CHAR(112,"00000000
00000000",113,"0000000000000
000",114,"0000030F1F3F7F7F",
115,"7F7F7F7E7C391103")
1010 CALL CHAR(116,"00000000
00000000",117,"01030307070F1
F3F",118,"070F1F3F7F7FFFFF",
119,"FFFFFFFFFFFFFFFF")
1020 CALL CHAR(120,"FFFFF7E7
CF9F3F7F",121,"FFFFFFFFFFFFF
FFF",122,"FFFFFFFFFFFFFFFF",
123,"FFFFFFFFFFFFFFFF")
1030 CALL CHAR(128,"01010101
01110909",129,"09080C0405020
000",130,"0000000000004020",
131,"108844FF42210000")
1040 CALL CHAR(132,"00301D0F
1622528A",133,"060908FF80000
000",134,"8080008080402000",
135,"088A96D421400000")
1050 CALL CHAR(136,"00000000
00000000",137,"0000000000000
000",138,"000000005000C040",
139,"5088829084A090A2")
1060 CALL SPRITE(#28,96,2,41
,145):: CALL SPRITE(#27,100,
2,73,145):: CALL SPRITE(#26,
104,2,105,145):: CALL SPRITE
(#25,108,2,57,113)
1070 CALL SPRITE(#24,112,2,8
1,81):: CALL SPRITE(#23,116,
2,113,81):: CALL SPRITE(#22,
120,2,89,113)
1080 CALL COLOR(12,2,2):: FO
R COL=15 TO 19 :: CALL VCHAR
(16,COL,124,3):: NEXT COL ::
 CALL HCHAR(18,19,124)
1090 CALL SPRITE(#10,128,13,
121,97):: CALL SPRITE(#11,13
2,13,121,129):: CALL SPRITE(
#12,136,13,89,129)
1100 SUBEND
```

Chicago TImes AUGUST 31st 1987

©1987 Chris Bobbitt

Graphics

Converting GRAPHX to TI-Artist

I recently read an article in the Hoosier User's Group's excellent user group newsletter on converting GRAPHX to TI-Artist with interest. I too had faced this dilemma some time back in trying to transfer our popular GRAPHX Companion series of products over to TI-Artist (a project which regretably still has never been completed due to time limitations and a rather low priority). In any case, I thought the procedure we worked out may be of interest as well.

After playing with both programs for a while, we hit upon the solution offered by Mr. Robert Coffey. As a matter of fact, on our now discontinued Artist Companion disk there is a font that was converted over in just such a manner. It was so time consuming that we soon gave up.

The matter stayed dropped until 9 months or so ago when we were preparing Font Writer for release. I asked Peter Hoddie, the author, if he knew of a way to convert files over, and he said that Font Writer could be used.

Later that week, I sat down, and 4 hours later I had my first font. I chose a very elegant Times Roman, with a complete upper and lower case alphabet, from GRAPHX Companion IV (which was then in the editing phase) for the experiment.

The process is rather simple, actually.

Step one involves getting the fonts to TI-Artist format. If they are stored in a clipboard, as our GRAPHX Companion series fonts are, this involves first pasting them onto a screen (leaving plenty of room between characters, and then saving that screen to disk. As mentioned in Mr. Coffey's article, it is a very good idea to use an empty disk for this.

Next, enter TI-Artist and select the conversions section. Convert the screen from GRAPHX to TI-Artist format (load it in as a GRAPHX screen and save it under TI-Artist).

Next, enter the TI-Artist section from the menu, and select disk options. Load the screen into memory. Leave TI-Artist and go to the Enhancements option. The screen you loaded in TI-Artist will be in memory. Next, enter the Slides menu and select the Save Instance option. The filename you give should be the ASCII character that the picture you are saving represents (IE, if the picture is of an "A" the filename should be A). Next, the screen will appear. Move the cursor to the upper left corner of the character picture you are saving, press the fire button, and move it to the lower right side of the picture, boxing in the character. Press the fire button again and the picture will be saved to disk. Do this over and over until the whole font is saved as individual instances.

When you are done, exit TI-Artist and load your copy of our Font Writer program. Enter the Font Editor option.

When that portion of the package is loaded, go to the menu options, and select the option for opening a font for output. Do NOT select the Append font option. Next, enter the Instances selection from the same menu, and load in the first character (A or whatever). After it loads, you will be dropped to the graphic window.

(2022 note: Chris Bobbitt, the author, was the sole proprietor of Asgard, seller of Font Writer.)

At this point, it is a good idea if you establish a "baseline" first. The baseline is the bottom line that you will use (not physical) for placing the characters. Characters with descenders (which are the hardest to center), can be easily line up if you adjust them according to that line (remember such characters make look odd, what with all that empty space above them, but that is the way they should look - TI-Artist and Font Writer look at a font from the upper left hand corner).

Using the Move Picture keys of the editor, you can easily move the picture left, right, up and down to center it on that imaginary line (use the block boundary markers to avoid confusion). After the picture of the character is centered, enter the menus again, and again select the font options. Then select the option to save a picture in a font. The Editor will ask you what ASCII character it represents, and then a white cursor will appear on a screen showing the picture. Position this cursor, with the arrow keys, on the lower right corner and press Enter. The program will automatically save it to disk in the font file you specified as the ASCII character you specified. Do this over and over until all your Instances are converted to a font.

The advantages of this system over the one mentioned by Mr. Coffey (which, while still a good system and not requiring Font Writer) is that it is much faster, and you can center the characters in the font a lot easier since Font Writer's editor has tools for it.

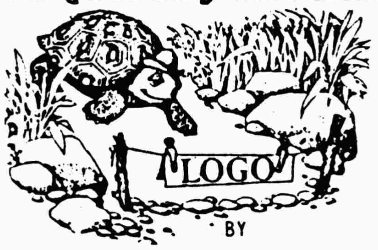
Converting regular clipart from GRAPHX to TI-Artist, of course, is a much simpler procedure since all you really have to do is paste it on a screen, convert the screen to TI-Artist, and save each individual picture as an Instance or Slide.

The reverse process, converting TI-Artist to GRAPHX, is very simple. All you have to do is get whatever you are converting onto a screen, save it to disk, convert the screen to GRAPHX (again using TI-Artist's conversions utility) and then paste it into a clipboard from a GRAPHX screen.

Regarding the legality of converting art from GRAPHX to TI-Artist; I'm not sure what the policy of other manufacturers is, but ours is that once you buy the stuff, it's yours. You can convert it to any format you like. However, remember that the works in our GRAPHX Companions and Artist Instances series ARE copyrighted (they are in fact the product of literally thousands of hours of work - a single font may take up to 10 hours to draw with GRAPHX!), and you can't give them out to anyone else. You can convert them for your use, but no one elses.

Font Writer is also copyrighted to J. Peter Hoddie and is manufactured and distributed by Asgard Software. The use of it as described here is only one of the many functions of the product.

LOGO (Turtle) TUTORIAL



BY

MIKE SLATTERY

It is a nuisance to have to keep retyping or just editing a procedure to get a different result.

LOGO has made things easier by allowing inputs, similar to BASIC or XBASIC. Naturally the commands are different. The general form of procedures with inputs is shown below:

TO (name) :variable 1 :variable 2 and so on.

Note: it is essential to include
A) a colon (:) in front of your variable name
B) a space in front of every colon in the title line and subsequent procedure name used
C) no space between the colon and the variable name being used.

If no space is left in front of the colon LOGO assumes it to be part of the procedure name. Thus BOX :A is a procedure name with input A whereas BOX:A is simply a procedure name with no inputs. If you type the latter, then try to input a variable you will get an error message.

The LOGO manual is not well set out as it appears to show no space before the colon(:). I spent a lot time before finding that the space is essential.

Also when typing in the procedure name do not leave a space between the colon and the variable name or LOGO will not recognise that name as that of a variable.

To see how procedures with inputs work try:

```
TO BOX :SIDE
REPEAT 4 [ FD :SIDE RT 90]
END
```

Now type BOX 40 and ENTER it. A box of side 40 is drawn. Without clearing the screen enter BOX 60. A larger box will be drawn. Boxes of any size can be drawn using the above procedure.

LOGO permits the use of more than one variable in procedures. Try:

```
TO RECT :LENGTH :BREADTH
REPEAT 2 [ FD :LENGTH RT 90
FD :BREADTH RT 90 ]
END
```

Enter RECT 30 60 and a rectangle will be drawn. RECT 60 30 draws the same rectangle but on its side. Any combination of length of sides can be used. If the same value is used for both length and breadth a square will be drawn.

The angle can also be a variable:

```
TO ANG :SIDE :ANGLE
REPEAT 4 [ FD :SIDE RT :ANGLE ]
END
```

In this example only 90 for the angle will give a regular closed figure, but try and see what you get with different angles.

The following procedure will draw a closed figure:

```
TO CLANG :SIDE :ANGLE
REPEAT 2 [ FD :SIDE RT :ANGLE
FD :SIDE RT (180- :ANGLE) ]
END
```

This will draw a figure with equal length sides. Parallelograms with sides of different lengths can be drawn with the following:

```
TO PARA :SIDE1 :SIDE2 :ANGLE
REPEAT 2 [ FD :SIDE1 RT :ANGLE
FD :SIDE2 RT (180- :ANGLE) ]
END
```

The above procedure can be made as complex as you wish by adding SIDES and ANGLES as necessary. But to get a closed design, all angles must total 360.

Inputs can be used in all normal arithmetic operations, i.e. they can be added, subtracted, divided or multiplied by either constants or other inputs.

The following examples will show how this is done, and will draw some very interesting designs at the same time.

EXAMPLE 1.

```
TO SQ :S :A
CS SXY 0 25
L :S :A
END
```

```
TO L :S :A
FD :S RT :A
L :S+1 :A
END
```

S is the initial length of the side of the design and A is the angle. The length increments one unit each cycle until out of ink. Angles of particular interest are 60, 90, 120, 135, 157 and 210. Of special interest are the following two groups: 117, 120, 123 and 117, 180, 183.

Run the procedure by entering eg. SQ 1 90. The order of entry of the variables is important. The angle must be after the side otherwise LOGO will take the first variable to be the side length.

Note: for clarity :S+1 can have brackets ( ) around it. This will be of more use when more than one variable is being altered at the end of a procedure. Putting them into brackets is not essential to the running of the procedure.

EXAMPLE 2.

```
TO SQ1 :S :A :IN
CS SXY 0 25
L1 :S :A :IN
END
```

```
TO L1 :S :A :IN
FD :S RT :A
L1 ( :S + :IN ) :A :IN
END
```

:IN is the incremental value and can be any number. If you want to vary the angle, you can use either a fixed increment to give

```
L1 ( :S + :IN ) ( :A + :IN )
```

if you want to increment the angle by one unit per cycle or

```
L1 ( :S + :IN ) ( :A + :P ) :IN :P
```

if you want to use a variable for the increment. If you use a fourth variable in the preceding example, do not forget to include it in the procedure title line and also in the return line at the end of the procedure.

To finish, here is a program to demonstrate the graphic capability of the turtle mode:

```
TO A
CS SXY -60 -10  A1
FD 40 RT 10 FD 55 LT 20 BK 55
FD 30 RT 100 FD 72 RT 65 FD 34
SXY -50 -10 RT 35 FD 37 BK 37
RT 170 FD 20 RT 90 FD 10 RT 90
FD 20 LT 10 FD 37 SXY -10 0
REPEAT 3 [ A2 ] SXY 110 -13
REPEAT 10 [ A3 ] PU LT 85 FD 15
PD REPEAT 4 [ A3 ]
END
```

```
TO A1
REPEAT 2 [FD 40 RT 90 FD 100
RT 90 ]
END
```

```
TO A2
SH 70 FD 10 RT 90 FD 10 RT 30
FD 10 RT 120 FD 10 RT 30 FD 10
RT 90 PU BK 20 PD
END
```

```
TO A3
A4 PU RT 60 FD 10 RT 90 FD 30 PD
END
```

```
TO A4
SH 30 REPEAT 4 [ A5 RT 180 ]
PU RT 230 FD 7 RT 10 PD FD 5
RT 10 FD 10 RT 20 FD 10
END
```

```
TO A5
REPEAT 10 [ FD 1 RT 27 ]
END
```

"A" runs the program.

TI Sydney Users' Group

LOGO 1 OR LOGO2

C L A S S I F I E D S

MINIMEM Conversion to Rechargeable Battery. Send MINIMEM and crossed cheque for £7.50 to N.J. PETRY, TENSAL TECHNOLOGY, 15 PENRICE CLOSE, WORLE, W.S.M., AVON BS22 9AH.

FOR SALE:

Outfit 1:

TI-99/4a computer with manuals, TV modulator, Cassette Recorder and TI leads (including 15 cassettes - some with programs), TI Thermal Printer with 6 rolls of paper (and manual), TI Invaders and TI Tombstome City modules, TI Speech Synthesiser (with manual), Flight Simulator cassette and "Getting to know the 99/4a" by Stephen Shaw.

Complete package: **£150.00p**

Outfit 2:

TI-99/4a computer with manuals, TV modulator, Peripheral Expansion Box with the following already fitted; 5¹/₄" Single sided disc drive, 32k Memory Expansion card, Disc Control card, RS232 card and Flexible Cable Interface. Complete with manuals for each item. Also included TI Extended Basic (with manual), TI Writer (complete with Disc and manual), TI Disc Manager II module, Terminal Emulator II module, TI Graphing Package cassette and 30 once used 5¹/₄" floppy discs in library boxes.

Complete package: **£350.00p**

The above items are in excellent condition, personal reasons forces sale. Near offers would be considered and delivery can be arranged by mutual agreement.
Packages could be split up providing the majority of the outfit was required.

Please contact: John R.Waters
01.997-5200 after 7pm.

Console,power pack & modulator,joysticks,ExBas,12 modules,5 books.
£75, willing to split. R.P.SKUSE Tel. 040 923 311.

TI99/4A £35, SPEECH SYNTHESISER £25, TI STARTER PACKS 1 AND 2 £5 EACH, TI GAME WRITER PACKS 1 AND 2 £5 EACH.

VARIOUS GAMES AND BOOKS, PLEASE WRITE OF PHONE FOR DETAILS, TO COLIN C. STREDDER, 84 GREENWAY ROAD, BIRKENHEAD L42 6QS.
051 644 0520 AFTER 8.00PM.