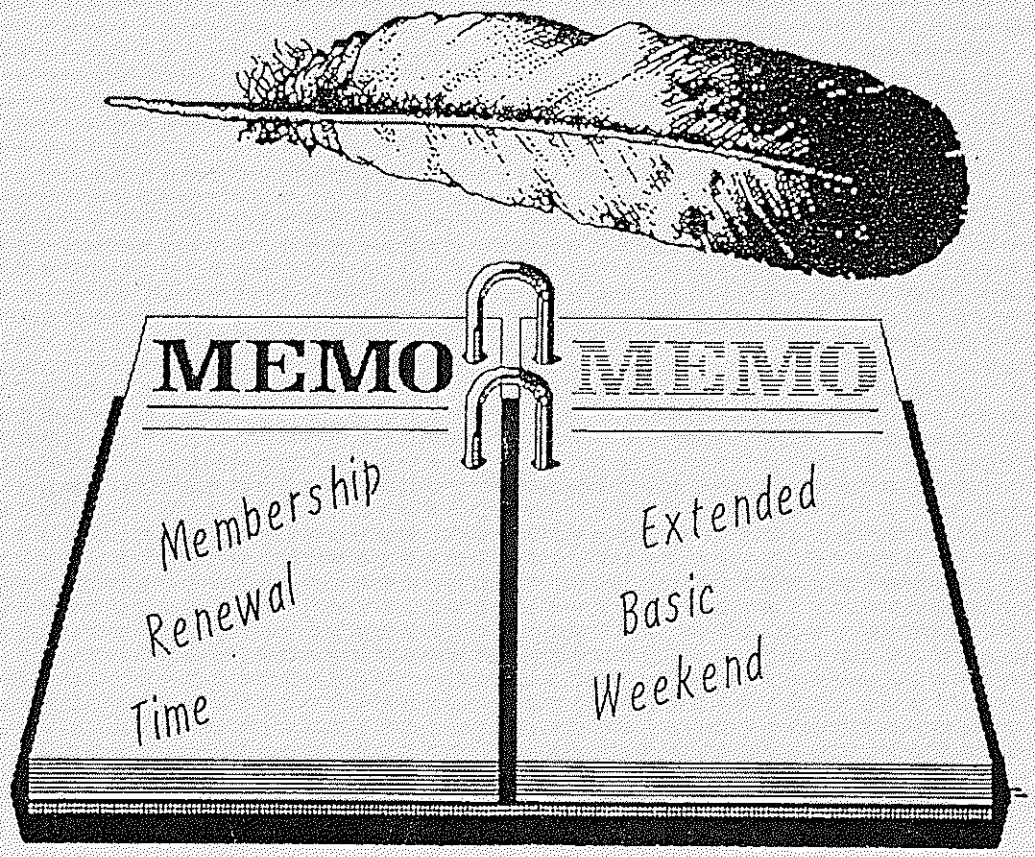
TIsHUG NEWS DIGEST

Focusing on the TI99/4A Home Computer

Volume 12, Number 3 April, 1993

Registered by Australia Post - Publication No. NBH5933



Sydney, New South Wales, Australia $3

TIsHUG (Australia) Ltd.
A.C.N. 003 374 383

TIsHUG News Digest

April 1993

All correspondence to:
C/o 3 Storey St.
Ryde 2112 Australia

TIsHUG Sydney Meeting

The April Meeting will start at 2.00 pm on the 3rd April at Ryde Infants School, Tucker Street, Ryde.

Printed by
Kwik Kopy Parramatta

TIsHUG News Digest ISSN 0819-1984

I N D E X

Notice

MEMBERSHIP RENEWALS NOW DUE !
REFER TO MAILING LABEL

Editor's Comments

by Bob Relyea

We are already well into the year and the response from the membership for local articles has been generally good, so far. Thank you to those who take the time to write these nice articles that we all enjoy and benefit from. I hope you take the time to read the article that Alf Ruggeri put together which, in part, spells out what he intends to do for the All Day Tutorial in May. At least there is one user who thinks ahead! If anybody else has something definitely lined up for the May tutorial then I have to know and have the material within a week so it can make the May magazine. This will give the membership an idea of what is on for the day so that plans can be made. If I am not required to do something else I plan on having my system there (which includes Funnelweb on Eproms on the Mini/system Ramdisk) to do a bit of programming or whatever is required. More details in the May issue. The new paste-up committee is off to a good start and learning quickly. Thanks for your efforts guys.

BITS AND BITES PT. 1

Typed and Reviewed by Larry Saunders

TI 99/4A Pole Position

AND THEY'RE OFF

Install your engine

Load the POLE POSITION game from the ROOT loader. The title screen should appear. Now press any key to display master selection list.

Choose one of the three difficulty levels and/or the number of laps in a race(1-8).

Press ENTER to start the game. Press "P" to pause during game play and any other key, joystick, or fire button to reactivate the game after pause.

NOTE; Make sure that the ALPHA LOCK key is in the up position when using joysticks.

The object of Pole position is to pit yourself against the clock and the competition (other high-performance racers). Now is your chance to prove you have got the nerve and skill to be a professional race car driver. You can steer by moving the Joystick to the left or the right. Increase or decrease your speed by moving up or down, respectively. Pressing the "FIRE" button allows the driver to change from low to high gear. In the extreme upper right-hand corner of the playing screen, is your Lap Time Counter (0"00). The Time Clock at the center, which determines the remaining time your car has to cross the finishing, winds down as your Lap Time Counter increases. Also, at the top right is the high and low gear indicator (HI/LO). Your speed as shown can reach a maximum of 195 MPH.

KEYBOARD CONTROLS

	Left Handed	Right Handed
Increase Speed	E	I
Decrease Speed	X	M
Left	S	J
Right	D	K
Gear Shift	Y or	Q or V
Upper-Left	W	U
Upper-Right	R	O
Lower-Left	Z	N
Lower-Right	C	,

When the game screen first appears you are ready to qualify for a race. Before you can compete in any races, you must qualify for one of eight starting positions. you have 90 driving seconds in the qualifying run, but must achieve a time lap of 73" (seconds) or better to qualify for a race that was selected begins in a matter of seconds.

The first lap in a race has a maximum time limit of 75 seconds. More cars appear on the course in each successive lap. But you begin a new lap with additional time to complete it - only if you have finished the lap in the allotted time.

If you hit an other car, your car will explode, costing you precious time. You will also wipe out if you run into a road sign. No matter how many times you crash you will receive another car until your time runs out.

Part of the playing strategy is to try and keep your car on the road - you lose time (and thus points) when the car is off the track. Skidding also causes your car to slow down. Gun it in the straights. If you find yourself going to fast reduce speed to ease around the more difficult turns (gear shift to low).

STRATEGY TIPS

*...Always shift from low gear to high gear around 100 MPH for best acceleration.

*...Try to stay on the center stripe line most of the time. You can squeeze between two computer cars in this position.

*...The beginner should play the game with his car at the lower speeds not 195 MPH!

*...For a double braking effect when needed. slow down by reducing your speed and simultaneously shifting to low gear.

*...If the cars immediately ahead of you are packed tightly together, wait for them to separate before attempting to pass.

*...If no billboards are obstructing your path, you might be able to pass other cars by driving on the grass. (with a lot of LUCK).

*...Try to take the curves with a series of small turns rather than one large turn. as this will reduce skidding which slows your car.

NOTE: Not compatible with Version 2.2 System.

SCORING

Playing Pole Position, you score in several ways, as shown below:

Every lap completed is worth 10,000 points.

Each car you pass is worth 50 points.

After you cross the finish line, each second of time left on the Time Clock is worth 200 points.

The chart below lists the qualifying lap times for the eight starting positions in the race, and the number of bonus points awarded for each qualifying time:

Starting Position	Lap Time	Bonus Points
1	58"50	4000
2	60"00	2000
3	62"00	1400
4	64"00	1000
5	66"00	800
6	68"00	600
7	70"00	400
8	73"00	200

TI BITS NO.22

By Jim Swedlow

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

I had a problem recently making TI Writer print out some revisions to our club bylaws and I though you might be interested. Some background is needed first.

TALKING TO YOUR PRINTER

Some time ago I mentioned that there are almost no standards for software codes to control printer functions (font, appearance, line spacing, etc). There are, however, a few things that are uniform. This month's effort centers on two of them: carriage return and line feed.

For simplicity, I will refer to them as <CR> and <LF>. <CR> is ASCII code 13 and <LF> is 10 (or, in hex, OD and OA, respectively).

What do they do? Just what they say. A <CR> causes your print head to return to the beginning of the current line. A <LF> causes the print head to advance one line, in the same print position. Sending both <CR><LF> causes the print head to move to the beginning of the next line.

This is almost universal. Many printers, however, have the ability to add a missing <CR> or <LF>. When in this mode, receipt of either command causes the print head to move to the beginning of the next line. Receiving both causes a double space. Usually this is controlled with DIP switches and is discussed in the printer's manual.

When you open a printer as "PIO" (I will not discuss RS232 names here because they act just the same), your RS232 card sends your printer a <CR> and a <LF> after every print line. You cannot control this as it is automatic.

You have some software control, however, by adding ".LF" or ".CR" to the printer name.

Here is a simple program that will show you what these extensions do:

```
100 DATA "PIO", "PIO.LF","PIO.CR"
110 FOR I=1 TO 3
120 READ A$ :: OPEN #1:A$
130 PRINT #1:A$:A$
140 CLOSE #1
150 OPEN #1:"PIO"
160 PRINT #1: : :
180 CLOSE #1 :: NEXT I
```

When the printer name is PIO, your RS232 card will add a <CR> and <LF> to each line and your printer will print:

```
PIO
PIO
```

When the printer name is PIO.LF, you RS232 card will only add a <CR> so you will get:

```
PIO.LF
```

but it will be printed twice (bold). With PIO.CR, neither <CR> nor <LF> is sent so you should have:

```
PIO.CRPIO.CR
```

PRINTERS AND TI WRITER

Wait, you say, in TI Writer (or FUNNELWEB, etc), I can add <CR> to lines. Does this change things? The answer is yes and no.

The primary function of this <CR> is to tell TI Writer that you have reached the end of a paragraph. It uses it in the Editor when you cause a reformat and in the Formatter when you use FIll or FIll and ADjust.

What does it do to your printer? Well, lets say you have this text in your Editor:

```
UGOC<CR>
```

and then print it with PrintFile and PIO as your printer name. Your TI Writer would send this to your printer:

```
UGOC<CR><CR><LF>
```

In other words, it adds a <CR><LF> to whatever is on the line. The double <CR> is the same as a single <CR> as the print head can only move to the beginning of the print line once.

WHY DOES THE FORMATTER USE A ".LF"?

In short, the answer is to enable you to use bold (@) and underscore (&). Remember, when your printer name is PIO.LF, your RS232 card only adds a <CR> to each print line. Here is how it works.

If your entire line was:

```
@UGOC
```

The formatter would send the following to your printer:

```
UGOC<CR>
UGOC<CR>
UGOC<CR><LF>
```

This would cause triple printing of UGOC or bold. Underline is similar. If your entire line was:

```
&UGOC
```

The formatter would send the following to your printer:

```
UGOC<CR>
_____<CR><LF>
```

On the page, UGOC would be underlined.

STRIKE OUT AND ITALICS

I am almost done with the background. As I said at the beginning, I was working on an update to our bylaws. The normal way to show changes is to strike out old text and put new text in italics. For italics, I used TransLiterate to change the open bracket "{" to start italics and the close bracket "}" to stop italics:

```
.TL 123:27,52
.TL 125:27,53
```

Whenever the Formatter runs into an open bracket "{" or ASC 123, it will send the ASC codes 27 and 52 to the printer, which switches to italics. Similarly, "}" or 125 causes the Formatter to send 27 and 53 which returns my printer to its normal font. It made editing easy as anything I wanted in italics I simply enclosed in brackets like {print this in italics}.

For strike out, I used transliterate to change the underscore to a dash:

```
.TL 95:45
```

Then, when I used the underscore command, the Formatter printed strike out. So &old &text would print as:

~~old text~~

NOW FOR THE PROBLEM (FINALLY)

When I mixed strike out and italics on the same line and the italics were first, I got garbage. Why? Because TI Writer and my printer count characters differently.

Suppose I had this line:

(new text) for &old &text

When it went through the formatter, the printout looked like this:

new text for old ~~text~~----

Why? Well, you see, TI Writer counted the open and close brackets as characters while the printer, recognizing them as control codes, did not. So TI Writer thought it was sending this:

..new text.. for old text<CR>
--- ----<CR><LF>

The double dots ".." stand for the control codes that the brackets were translitterated into. TI Writer, as you can see, counts the number of characters sent to the printer, not the number in the Editor print line.

The problem was that the printer counts the print characters this way:

new text for old ~~text~~----

The result was that the strike out was not correctly aligned.

The solution was to make sure that, in the same line, italics did not precede strike out. It could follow or be on a different line, but not precede.

So I wrote this:

&Old &Text becomes (new text)

and it printed like this:

~~Old Text~~ becomes *new text*

I left ADjust off (using only FIll) because this difference in character counts defeats the way the Formatter right justifies text.

********** END OF ARTICLE **********

TREASURER'S REPORT

by Cyril Bohlsen

It is now time for the majority of our members to renew their membership. Please look at your mailing label to see when your subscription expires.
IF YOU ARE UNFINANCIAL THIS WILL BE THE LAST TND YOU WILL RECEIVE.

Income for March	$ 1394.40
Expenditure for March	$ 1369.55
Profit for month	$ 24.85

THE TEACHING COMPUTER

by Jim Peterson

I still consider the TI-99/4A to be a HOME computer, although many users have expanded it far beyond that.

And what are the uses of a home computer? Primarily, entertainment and education - and, if you can justify the expense of an RS232, printer and modem, word processing and telecommunications.

The importance of the computer in education has been overemphasized in TV advertising, to the extent that people were offended by suggestions that, if they failed to buy a home computer, they were condemning their children to a life of failure. One man even formed an organization to oppose the TI-99/4A advertisements !

However, the educational potential of the TI-99/4A was never realized. To teach any subject, a planned series of lessons is required. And, since each lesson is soon learned and no longer needed, the individual lessons must be inexpensive.

Texas Instruments did put out several educational modules, but they were not in any planned series and, in those days, they were not cheap. The Plato series is carefully planned, and excellently designed from an educational standpoint - but it does not take full advantage of computer capabilities, and is far too expensive.

Third party manufacturers did put out some fine educational modules but again, no planned series, and I do not know of any of them still supporting the TI.

In the early days, Micro-Ed and a few others put out some good educational software on cassette. As far as I know, Kidware and Tigercub are the only ones still offering educational programs on cassette.

There is also a vast amount of public domain software written by amateur programs. It ranges in quality from mediocre to excellent, consists mostly of pre-school teaching or basic maths drills, and would take much effort to organize into any course of education.

The market for educational software has been so poor that it is doubtful that any more will be produced. So, if you need it, you will have to write it for yourself!

Ideally, educational programs should be written by teachers, because they know how to teach. Unfortunately, it seems that few teachers have learned how to write programs, although some of their students have. A cooperative effort could have resulted in a large pool of good educational public domain software available to schools. However, the educational establishment as a whole has been so brainwashed by the Apple peddlers and the disciples of Logo that they will give no support to anything else.

What makes a good educational program? If possible, it should be interactive - it should: 1. teach a lesson; 2. test to see if you have learned the lesson; 3. if you have not, go back and teach it again; 4. if you have learned, go on to the next lesson.

This is not always practical in a single program, but can be done by having one program run the next program. I used a variation of this technique in "Casting Out Nines", which teaches a method of checking long multiplication and division problems. The student must correctly solve several problems at each step before the program will continue to the next step.

The program should give some kind of spoken, printed, graphics or musical reward for correct answers, and a corresponding admonition for wrong answers. The nature of these depends on the age level of the student. In "Kindermath" I use a groan and a frowning face which can be changed to a smile and music by a correct answer. In "Kinderminus", the problem is displayed in the center of a multicoloured kaleidoscope which changes patterns for each correct answer or turns black for a wrong answer. For an older student, a short game to be played could be offered after successfully answering a series of questions.

Computer graphics and sound should be fully utilized, but not allowed to become boring. The same musical salute after each correct answer soon gets tiresome. In some of my programs, a note is added to a tune for each correct answer, and after several such answers the entire tune is played. "Kindermath" uses several nursery tunes in succession.

The "stupid computer syndrome" should be avoided. This occurs when random selection causes the same question to be asked twice in succession. It is easily prevented by the simple statement IF Q=Q2 THEN (go back for another selection) ELSE Q2=Q.

The computer is especially well adapted to teaching maths, because it can generate an infinite number of random problems to be solved. When the problem requires keying in an answer to a multi-digit problem displayed on the screen, input should be accepted from right to left in the same way as it was being worked on paper. My " Maths Homework Helper" does this with addition, subtraction and multiplication problems of any size, and also helps the student by refusing to accept an incorrect digit.

When possible, if a wrong answer is given the student should be shown how to work the problem. In one of our Extended Basic classes we analyzed a maths quiz program which generated random problems in the form of "IF 3 BOYS CAN CATCH 12 FROGS IN 4 DAYS, HOW MANY FROGS CAN 9 BOYS CATCH IN 8 DAYS?". If the answer was wrong, a screen display explained, "NO, THAT'S WRONG. IF 3 BOYS CAN CATCH 12 FROGS IN 4 DAYS, THEN 3 BOYS CAN CATCH 3 FROGS IN ONE DAY".....etc., through the problem. Similar routines could be written for a wide variety of time/speed/distance problems, etc.

The basics of music education can also be easily taught by computer. The TI-99/4A can generate any musical tone required, and the piano keyboard, guitar fingering, musical notation, or whatever can be graphically displayed. John and Norma Clulow, Regena, and Bob Pomicter have written some excellent programs of this type.

Educational programs requiring much text are more difficult because of the limited memory capacity of the computer. Many of the public domain programs of this kind seem to have been typed in directly from a textbook, and there is really little reason for computerising them. Some good module software is also of this type - nicely programmed, but very soon learned and discarded.

Most speed reading programs on the market are loaded with a data base of sentences, which are flashed on the screen briefly and the student is then asked to repeat them. He soon begins to recognize them from memory, even though he may think he is reading faster. Also, the purpose of speed reading is to grasp the meaning of a sentence, not its exact wording. My "Speeder Reader" and "Junior Speeder Reader" programs bypass these faults by being loaded with a wide selection of individual nouns, verbs, adjectives and modifiers which are randomly selected and combined into an infinite number of sentences. Then, the student is asked any one of several different questions about the sentence, requiring a one-word reply.

Public domain spelling programs are popular but not very practical. In order to tell the student to spell a word, it must be either printed on the screen or spoken by the speech synthesizer. In the former case, he is already being shown how to spell it. The vocabulary of the speech synthesizer is rather small. The Terminal Emulator II permits an unlimited vocabulary, but the pronunciation is not very clear - a sentence may be understood, but a single word out of context is more difficult. Ron Binkowski published a "Speller" program in the 99'er which overcame this problem by allowing a separate phonetic spelling of the word and an "as in" phrase which could be phonetically spelled until it sounded right.

In my "Miss Spell", I programmed each word in a correct and incorrect spelling, randomly showed either one to the student and asked him if it was correct and, if not, to spell it correctly. In "I & E Spelling" I programmed every word containing those difficult "ie" and "ei" combinations, randomly showed them on the screen with those two letters replaced by blanks, to be filled in. If the answer is wrong, the student is shown a screen displaying the "I BEFORE E EXCEPT AFTER C" rule with all its exceptions.

Finally, the best educational programs of them all are those that teach a person something while he thinks he is just having fun. Word games are of this type, whether the popular "Hangman" or "Scrabble" or whatever.

My "Tirkle" is just a very simple little game for children, based on the early computer game called "Hurkle". However, teachers have told me that they find it very useful for teaching young children logical thinking and compass directions.

The possibilities are endless - and so little is being done!

********** END OF ARTICLE **********

GIF—MANIA TO PAGE PRO

by Alf Ruggeri

In the preparation of my Greeting Cards presentation/demonstration for the TI-Faire, I spent a total of 80 hours in processing GIF images and converting the same to PAGE PRO format. I had intended to produce several articles and full day tutorials to document the experience as full utilization of GIF-MANIA was realized, however when Ross Mudie rang me on the 2.1.1993 to inform me that he had succeeded in producing a TEXPAC downloadable GIF file format, he caught me completely off balance.

I have decided to produce the articles earlier than I had intended but unfortunately because the time used for the TI-Faire preparation, generated a monumental backlog of domestic duties, coupled with several scheduled work field trips, I will not be able to immediatley proceed with the articles task. However as soon as time permits I will do so.

The articles employ several TI-screen & PAGE PRO-screen area maps and co-ordinates tables mandatory to the method used which will limit their availability to the hardcopy format only, regrettably not TEXPAC downloadable.

********** END OF ARTICLE **********

BITS AND BITES

typed by Larry Saunders

Beginner's Corner, Part 2

The main feature last month, "The NEW 99/4A", is really the continuation of this article from the March Bits and Bites issue. However, we are far from running out of objects of discussion. This month we are going to discuss database concepts.

One of the most popular uses for a computer (only after word processing) is "DATABASE MANAGEMENT". I put this in quotes for very good reason - this term is often misused, rarely defined, and has been so maligned for so long it is practically meaningless. However, there IS a definition for it. Database management is the practice of managing (running, or otherwise controlling) a collection of related data. Data itself can be short lengths of text, number, prices, or really any little bit of knowledge that you would like to store on a computer. Data can be anything from the stuff you write on a PD99 form to the birthdays of your friends and relatives to their names.

A database, as I said, is a collection of related data. What do I mean by "RELATED?" Well, for instance, a single database might contain the names and addresses of your friends. It will most likely NOT contain the pedigrees of your dogs as well. In a database, as in life, things tend to be organised together. You COULD put the pedigrees of your dogs with the names and addresses of your friends (perhaps), but it hardly makes sense. More likely, you might want to have a list of your friends names and their phone numbers as well. That would be related data. We can make an assumption that the only purpose of putting data in a database in the first place is so that you can get it out later in some coherent fashion. Mixing different data together is not going to promote that. So, a database is simply a collection of data that seems like it "goes together".

A database program (often called, improperly, simply a "database"), is a program designed to "MANAGE" your database. What do I mean? Well, to be more specific, it will typically allow you to type data into a database, find items of interest in the database, perhaps sort the data in some order, and eventually, print out part or all of the data. There are many types of databases "Flat files", "relational" (and if you are an old timer), "network" and " hierarchal ". But, I am getting ahead of myself now.

As you might have guessed, a database program is usually actually many little programs, which together are often called a "DATABASE MANAGEMENT SYSTEM" (or, in Government-speak, a DBMS). DBMS can be as simple or as complicated as the programmer wants them to be. Some are so complicated that you need a very sophisticated "language" to talk to them in, you need to type in commands in an order that the systems understands in order to accomplish even the most basic tasks (like type data in). The advantage to something that difficult? Well, usually there is a trade-off between power and ease-of-use. Something that is really easy to use cannot do anything really sophisticated, and something really hard to use really is not worth using for anything simple, but is the only way to do some complicated stuff.

The two most common types of database management systems in the 99/4A and the Geneve world are "FLAT FILE" and "RELATIONAL". Actually, to be more precise, the most common type of 99/4A database is the "flat file" method. There are only two relational database management systems for the 99/4A (albeit, two of the more popular such programs). The other two types have actually never been developed for the 99/4A. There is little loss, though - a relational database is usually as capable as those two ways of managing a database.

What is the difference between the two? Well, hundreds of articles have been written on the subject, but it basically boils down to the way data is stored and retrieved. In fact, storage and retrieval are the entire reason FOR a DBMS - everything in a DBMS is somehow related to put data in the computer (or the database), and getting it out again. So, when we say that is the "only" difference, we paint it with a very broad brush.

Before you can understand the difference between these two systems, it helps to understand some more basic terminology. A database system can be viewed on three levels - the physical, the user's view, and the conceptual. The physical level is the way the computer views the database - it is how the computer actually stores the data in the database. The user's view is simply how the data is significant to you when you type it in, as perhaps a listing of all your computer equipment. The conceptual view is best understood as the halfway point between the two views - it is the view that both the computer and the user can understand. A typical user usually does not understand (and probably does not care) how the DBMS stores the data. The computer, of course, can hardly understand human thought processes or precepts. The conceptual level is the bridge between the two. It is typically the level in which the user interacts with the DBMS and the database.

The conceptual level is just that, an easy-to-understand concept of the database. A database is typically conceptualised as a "table". A database containing your friends names and phone numbers might look like this in a table.

Name	Phone Number
Dick	918 8132
Percy	808 3181
Peter	528 8775
Steven	608 3564

Fortunately, a computer does not have too hard of a time understanding what a table looks like either. Hence, both you and the DBMS understands the data.

Now let's go into what a database itself actually is. If you examine the table a little further, you will notice that it is in two columns - one labeled "NAME" and one labeled "PHONE NUMBER". In DBMS lingo, those columns are known as "FIELDS". A field is exactly like a column in a table. The rows of the fields are the data in the database (each individual row is known as a "TUPLE", pronounced "TOO-PULL"). Tuples are also called "RECORDS" in some database programs. When you create a database, you have to tell it what "FIELDS" you want, what type of fields they are (you will notice that the "NAME" field is all text, while the "PHONE NUMBER" field is all numeric), and how much space you should give each Once you have told the database this information, it creates on your database disk something called a "SCHEMA" (pronounced "SKEEMUH"). The schema is the "DEFINITION" of the database.

Once you have created the database definition (or schema), consisting of all your columns (fields) and their lengths and types,you begin the process of actually creating the database. This is the laborious part of the procedure where you actually type in all the data into the database. The part of the program where you type in the data in your database is typically called a "DATA ENTERY SCREEN". This is a bit literal - in the database world it is called a "FORM". Some sophisticated database programs will allow you to design the data entry screen, or form - even down to placing the fields exactly where you want them on the screen, and even the screen colours.

In some database programs, the "FORM" is made to be part of the "SCHEMA". As you type the data into the database through the form it is stored by the DBMS in another format that is more readily managed by the program. Again, that format is the physical view of the database, or the way the DBMS sees your data in your database.

After you have typed in a lot of records, or tuples into your database, you will probably want to do something with them. There are many ways to use the data. Most likely you will want to create a "REPORT" - a printout of part or all of the data. You might want the data in the "REPORT" to be sorted by one of the fields (remember, columns) in your report. You might want everyone who lives in the (918) area code. In some databases, you are limited in how you can access your data. More sophisticated databases have what is known as a "QUERY" capability - really some way to ask "QUESTIONS" of your database, such as "WHO ARE THE PEOPLE IN MY PHONE LIST NAMED FRED, AND WHAT ARE THEIR TELEPHONE NUMBERS?". Of course, most databases will not simply let you enter a sentences like that one. That is the way to get data from the database in the user's view. Usually you have to use a "QUERY LANGUAGE". This language is a semi-algebraic way of getting certain records from the database. For instance, to get data from one popular PC database you would type "SELECT ALL FROM MYPHONELIST WHERE NAME EQUALS FRED ". After entering that "QUERY", the database would list all the database records with the name "FRED" in them, and incidentally their phone numbers as well. Again, this is a conceptual approach. The database might convert the conceptual version of the query into something like "select, myphonelist, name=Fred". The conceptual way is considerably easier to understand.

In order to get the list of "FREDs" sorted, in some databases you have to make another table of just people named Fred, and them have the database sort it in alphabetical order. Some databases will let you combine this into one step.

As mentioned above, once you have entered data into the database you will want to create a report. Almost all databases will let you designate, to some degree or another, how you want selected records printed on the page - the order of the fields, etc. Some will let you draw lines and boxes around fields, even. This designation is called the "REPORT". Sometimes it too is saved as part of the schema, but usually not.

As mentioned above. There are several types of databases. The two found in the TI-99/4A and geneve world are the relational and flat-file. Before you can understand them. you should have a firm grasp of the concepts elucidated above. We will tell you the difference between the two in the next issue of Beginners Corner in *Bits and Bites*

********** END OF ARTICLE **********

BBS NEWS
from SYSOP

GIF files implanted in Extended BASIC programs are now on the BBS in the PROGRAM area. Full details on how it is done plus lots more on GIF may be found in the NEWS menu. The program GIF MANIA is also available for download.

The file METEX_I/F in NEWS shows how to interface a digital multimeter with serial data out to a TI99/4A.

********** END OF ARTICLE **********

TISHUG SHOP
with Percy Harrison

As it has been some time since I published a list of the hardware available through the shop I thought it was time to list them again. Should you require any of these items would you please ring me on 808 3181 at least two days before our monthly meeting so that I can bring them to the meeting.

On the matter of opening our club up to other computer users, we have only received one letter about this matter and the writer was strongly in favour of the proposal can we therefore assume that those people who have not responded have no objection to the proposal being adopted. The matter was raised at the February meeting and discussed at some length, both by the fors and againsts, after which a show of hands was taken which resulted in a clear majority voting for the proposal. The meeting closed with no immediate plans being made as to when and how any proposal should be instituted, however it was made quite clear by all in attendance that such a move would not in any way affect the current services offered to the TI computer club members and in fact should assist in maintaining such services for a longer period of time. Any comments regarding this proposal would be most welcome and should be directed to our Co-ordinator, Dick Warburton.

PRICE LIST.

5.25 in. DSDD Disks (Boxes of ten)	$6
5.25 in. HD Disks (Boxes of ten)	$10
3.5 in. DSDD Disks (Boxes of ten)	$10
5.25 in. DSDD Half Height Drive (New)	$65
12 Volt AC Transformer	$4
13 Volt Arlec Transformer	$12
8.5, 17 Volt Transformer	$25
60 VA Transformer	$20
MFC Printed Circuit Board	$30
MFC Kit (Disk Controller)	$103
Music Kit with PCB	$65
32K Memory PC Board	$7
Horizon Ram PC Board	$45
Horizon Ramdisk Basic Kit	$35
Funnelweb Eprom Set (3 Eproms)	$36
TI Artist Eprom Set (2 Eproms)	$24
32K Static Ram IC (62256)	$10
8K Static Ram IC (6264LP)	$5
Exchange Console	$30
ROS Version 8.14	$12
Mini Master 99	$70
Mini PE RS232/32k Card	$80
Mini PE RS232/PIO PC Board	$30
Mini-mem Battery	$3.50
Modulator UHF or VHF	$15
TI Power Supply	$15
TI 32K Memory Card	$40
RS232/PIO Card	$80
Modem PE Card (300 Bd)	$80
PE Expansion Box with I/F	$150
PE Ramdisk (184k Eprom)	$140
PE Ramdisk (248k Horizon)	$170
PE Ramdisk (248k Eprom)	$180
PE Ramdisk (320k Horizon)	$210
Printer (Serial)	$120
Thunderer Modem with Viatel Cart	$130
Standalone Disk Drive	$40

Packaging and postage extra on all items.

Bye for now.

********** END OF ARTICLE **********

SOUPED UP SUPERCARTS

by Andy Frueh, Lima UG

It all started with the original TI Cartridges. The cartridge ("cart") port of the TI is one of its advantages in both ease-of-use and young education. This port (called the GROM port) allows users to insert cartridges into its slot. This offers instant access to the programs contained in it. I mentioned that this port is an advantage. It is also now unique to the TI. Most other computers do not offer such a port and those that do not offer a whole lot of software. Older computers usually have such a port. The main reason for this is that home computers appeared shortly after the home video games (Atari, VCS, etc.) came out. Computer manufacturers feared that many people would be afraid of disks or cassettes, so they enabled their machines to use software in cart format, just like the popular home video game machines. A good marketing move. In addition, children could use them with considerable ease.

The quality of TI made cartridges grew from awful to excellent. The chief reason for this is the 99/4 was not as capable as the 99/4A. If you look at some later carts (1983), you will see that the manual may say, "for the 99/4A Home Computer only." This means they will NOT work on the 99/4. Most of these are games, such as Parsec and Star Trek. I am not sure if it was the graphics or the speech that would cause problems on the 99/4. Could somebody please submit a letter or an article describing the internal differences between these sister computers? Compare an early module or cart such as the 1980 Hunt the Wumpus to the 1983 graphic adventure Return to Pirate's Isle. The features and graphics of "Return" is significantly greater.

But what does all of this have to do with Supercarts? Well, just as TI tried to improve the quality of their cartridges, many users were "playing" with the cartridge and the GROM port it plugs into.

When most people think of Supercarts, they are thinking of cartridge-like hardware that plugs into the GROM port and have battery backed memory. Most Supercarts can save Assembly programs or even other cartridges and store them in this memory. However, when I use the term Supercart, I intend to discuss almost EVERY device available that can be plugged into this port.

Let's start from the beginning. One of the first Supercarts ever made was marketed by DataBioTics. This was the Superspace cartridge. This included an Editor/Assembler, it could save Assembly programs in its 8K memory and could save any cartridge that did not use TI's GROM (such as Atariasoft and Funware). What is a GROM? Well it stands for Graphics Read Only Memory. It is completely separate from RAM and ROM, which I assume most users are at least slightly familiar with. These are unique to TI. Three inside the console control the Operating System and BASIC language. Up to five addition GROMs (each with 6,144 bytes of ROM, I believe) can be added via the GROM port, adding 30K to the computer's ROM memory.

DataBioTics later marketed an upgrade to their Superspace. Superspace II had all the features of Superspace as well as letting users use 8K of memory as extra RAM for Assembly or TI BASIC programs. It also had 32K of battery backed memory, instead of only 8K. One thing I should note. Both Superspaces contained an Editor/Assembler. They did NOT include the manuals or support file disks. This may have been due to copyright restrictions. I am not sure if the operation of this Editor/Assembler is identical to TI's or not.

The next device I would like to talk about was the GRAMKracker. And no, I will not discuss other GRAM devices, such as the P-GRAM or GRAMulator. Not to take anything away from these two. All three devices perform similar functions. However, since the focus of the article is on devices that actually plug INTO the GROM port, I will only discuss the GRAMKracker.

I believe this came out sometime in 1986. At least that is the earliest advertisement I have seen for it. It was marketed by MG (originally Miller's Graphics). The main thing it could do was have another cartridge plugged into, then save the contents of that to a disk, RAMdisk or cassette file. You could also MODIFY your cartridges. A manual describing typical customising jobs was included. It also allows you to modify the TI operating system. For example, you could load in a new character set with true lowercase letters. Such changes would always be in effect, as long as the GRAMKracker was installed. It had 56K of memory, and could handle up to 80K. Obviously, this module was advanced over the Superspace.

What about prices? These are ORIGINAL prices, and I am sure these devices could be found at significantly lower rates. But here is how they used to compare.

Superspace.....$39.95 (This was a sale price. I cannot find it's original)
Superspace II.....$89.95 ($69.95 on sale)
GRAMKracker.....$174.95

The price of the Superspace dropped sharply after Superspace II was introduced. All three prices were around the same time, 1986-1987.

Those prices seem a little, well, pricey. And a lot of other users felt the same way. For that reason, pre-programmed and a few user-programmable "multi-carts" appeared. These were cartridges that contained many modules in one. I assume the idea for this started with the "Widget" or Navarone's module expander, which could hold three cartridges. The disadvantage was the size. This expander is fairly large.

One of the first multi-carts was the MultiMod by the late John Guion. His product was a plug-in modification to people who owned the Super Extended BASIC cartridge. This module was manufactured by Triton, which recently became TM Direct Marketing. I am not sure if TM offers this product. MG developed this cartridge for Triton. Super Extended Basic adds 33 new commands and 6 altered ones. Most of these are in the forms of CALLs. The Fall '89 catalog for Triton listed this module with a price of $49.95, including a graphics utility, Draw N' Plot. The MultiMod gave users Super Extended Basic, the Editor/Assembler, Disk Manager III, and TI-Writer in one module. The Multi-Mod was offered in kit form for $22.95 including manual and all support files for Editor Assembler and TI-Writer. I am not sure how well this sold, but I assume reasonably well. It gave users essentially the same thing as Funnelweb, without Funnelweb's enhancements and user expandability. The Multi-Mod's one major advantage was cartridge speed. Of course, the support files still had to be loaded from disk.

There is another similar product on the market, only this one is user-alterable. WAS Controllers offers a device which allows users to install 5 other modules in addition to Extended BASIC. For example, you could have Extended Basic, Editor Assembler, TI-Writer, Multiplan, Personal Record Keeping, and Tunnels of Doom in one cartridge. This is also sold as a kit, for $25. This includes the module, which has a modified case so all the other module's chips will fit. The flyer I have states that "only modules with 16 pin GROMs can be used with the Extended BASIC Expander". The GROMs are a normal looking computer chip. What that means is that you have to count the pins coming from the GROM chip. Only modules having 16 pins will work.

As mentioned in the December 1991 issue of B, B P, OPA is selling a $95 cartridge device with amazing features. It is called the Pop-Cart and looks about like any other TI cartridge. This device contains 256K chunks of RAM and GRAM. You tell OPA what modules or Assembly programs you want on the Pop-Cart, and they burn it into the module. This uses no batteries, but it does mean that users can NOT program it themselves.

Supposedly, if you put Extended Basic and the TE2 cartridge, the text-to-speech features of TE2 carry over into Extended Basic. This could have significant advantages, since many modules (especially some of TI's older ones) "add" commands to TI BASIC. You may specify up to 2 Megs of memory in 256K chunks for additional money, all in the same module. For an extra $25, OPA will also put their SOB (Son Of a Board) Operating System into the module.

What about weaknesses? Well this is the one aspect of TI cartridges that is rather disappointing. The connection between the cart and the GROM port can get very weak. This can cause some modules, especially Extended BASIC, to "lock up" The computer refuses to respond, and you usually have to turn it off.

There you have it. I believe that this is basically the principal "Supercarts" available. These devices prove the power and versatility of TI's GROM port. Many modern machines lack such a port. The users of these big machines claim that such a port is "infantile" and has no useful purpose. Bull! I feel this article PROVES how useful this port can be!

********** END OF ARTICLE **********

NUMBER SYSTEM BASE CONVERSIONS

by B. Takach

Newcomers wishing to explore and understand the working logic of computers will soon be confronted with different number systems. This brief and incomplete tutorial will try to shed some light on this often perplexing subject.

Inhabitants of our planet have adopted the decimal number system for the past few thousand years. You may happen to sit in a waiting room of a foreign consulate waiting your turn, and idly flip through the pages of a colourful periodical written in some strange language. The pictures will make sense, however the text will not be understood. Then some pages of statistical information may catch your eye. Now, the figures will be fully understood. Furthermore with some logical deduction, even the row and column headings will make some sense.

Financial data can be recognised by the currency sign, demographic data will show the /1000 or /100000 indicators. All these will be in the conventional decimal system, printed with the familiar arabic numerals. This was the happy state of affairs for many thousands of years until the end of the second world war. Computers were born around this time. Suddenly the time proven decimal system was no longer the top dog; the computer family could not use it.

Considering the birth of the decimal system, one could ask the question why has the base of 10 been chosen? Why not 5, 15 or 20? A simplistic answer claims it was due to being blessed with 10 fingers. Let us accept this answer, since without proof an argument would be fruitless, although mathematicians could put up volumes of convoluted arguments for and against the time tested decimal system. The fact remains we are stuck with it. Equally, it is also a fact that computers cannot function with it.

When you break down the complex computer to its itsy-bitsy basic building blocks, you will find these are nothing more than microscopic switches which have only two states: either on or off. Our first discovery is thus: The computer has only two fingers! Whatever problem it has to solve, the first task is to convert everything to the binary number system, then perform the requested task and finally convert the resulting data to the customary human conventions.

This applies equally to mathematical calculations, printable characters of the alphabet, sounds or music, drawings or graphics. Pictures of the most pleasing plump nude girl is seen by the computer as pages upon pages of noughts and ones. This is one of the reasons I am glad not to have been born a computer.

This incompatibility (here, I do not mean the nude blonde), forces us to work and think in some other number system when it becomes necessary to program a computer. Of course, as long as one is satisfied using one of the many high level programming languages the pain of working in a foreign number system can be avoided. However even then, the occasion may arise when the user has to step out of the decimal system.

As I said, the computer will ultimately work in the binary system. Unfortunately this is the least convenient system for humans. Firstly numbers very quickly become too long, secondly beyond the binary expression of decimal 10 they are almost impossible to remember. I was often astonished by early computer programmers, who could read and understand pages of binary code consisting of groups of 4 digits.

For this and other reasons - such as error recognition - A number of codes have been created besides the direct conversion from decimal to binary. Notable amongst these are the gray code and the BCD code. The most commonly used is the BCD code. BCD stands for binary Coded Decimal. It is an easily remembered code and a decimal number may be converted without pen and paper. The code is based on the fact that the width of the data bus and the processor capability is always divisible by 8, and 4 respectively. (Eg. 8bit, 16bit, 32bit micorprocessors)

We have to use 4 bits to write the decimal 9 in binary, thus each decade will need 4bits. Or in other words we can count from 0 to 99 using 1 byte (1 byte = 8bits). The binary number system and the BCD code is the same from decimal 0 to 9 however from decimal 10 onwards the binary number system follows the general rules of other number systems, whereas the BCD code will simply convert each decimal digit to the appropriate Binary number. For example:

Decimal	Binary	BCD
9	1001	1001
10	1010	00010000
99	1100011	10011001

There are other number systems used by the computer to aid human understanding respectively to make life easier. These are the octal and hexadecimal number systems. The octal number system would have been chosen by a race of mutants born without thumbs, the hexadecimal system on the other hand would be eminently suitable for four-armed mutants born without any thumbs. We have less difficulty understanding the octal system, as we have the appropriate number of digits in our decimal system to count up to 8. Let us have an example:

Decimal	Octal
0 - 7	0 - 7
8	10
10	12
80	120

It is unlikely that a TI-user will have to bother about the octal system, unless he/she is very keen to produce something very unusual. However the hexadecimal system will be encountered often.

Here we will run into a problem right at the very start: We do not have enough numeric symbols to represent 16 digits, we have enough for 10 only (eg.0-9). The remaining 6 are represented by the letters A-F. By convention these are capital letters, although most computers are not case sensitive when it comes to entering Hex numbers.

Finally how do all these systems and codes relate to each other ? Table 1 will shed some light on it. Whenever you are in the mood to experiment with conversions from one to the other, you may read up on the subject, perusing the reference list - these books are in the club library or do it the easy way, by typing in the conversion programs published in the listed references. Alternatively use the algorithm to write your own conversion program.

$$N = (X_n a^{n-1}) + (X_{n-1} a^{n-2}) + (X_{n-2} a^{n-3}) + \ldots + (X_1 a^0)$$

where:
N = the base 10 expression
Xi = the digits or terms of the base 'a' number
n = the number of digits or terms in the base 'a' number
a = the base in which the number is expressed.

The above formula may look frightfully complicated. Two examples will make it easier to understand.

1. We will convert the binary number 100011101 to decimal:

```
1 x 2 8  =  256 +
0 x 2 7  =    0 +
0 x 2 6  =    0 +
0 x 2 5  =    0 +
1 x 2 4  =   16 +
1 x 2 3  =    8 +
1 x 2 2  =    4 +
0 x 2 1  =    0 +
1 x 2 0  =    1
          -------
   N     =  285
  (dec)
```

2. Let us convert the hexadecimal number 11D to decimal:

```
1 x 16 2 =  256 +
1 x 16 1 =   16 +
D x 16 0 =   13    (D     = 13 )
          -------    (hex)
   N     =  285
  (dec)
```

References:

1. Understanding Computer Science, developed and published by Texas Instrument Learning Centre.

2. Understanding Microprocessors, TI Learning Centre.

3. Sourcebook for Programmable Calculators, TI Learning Centre.

4. TI 99/4A: 51 Fun and Educational Programs. by G.M. Schechter; Howard W. Sams Co. Inc. USA

5. TI-99/4A: 24 BASIC Programs. by C.A. Casciato D.J. Horsfall; Howard W. Sams Co. Inc. USA

Table 1. Base Conversions.

Decimal	BCD	Hex	Binary
00	00000000	00	00000000
01	00000001	01	00000001
02	00000010	02	00000010
03	00000011	03	00000011
04	00000100	04	00000100
05	00000101	05	00000101
06	00000110	06	00000110
07	00000111	07	00000111
08	00001000	08	00001000
09	00001001	09	00001001
10	00010000	0A	00001010
11	00010001	0B	00001011
12	00010010	0C	00001100
13	00010011	0D	00001101
14	00010100	0E	00001110
15	00010101	0F	00001111
16	00010110	10	00010000
17	00010111	11	00010001
18	00011000	12	00010010
19	00011001	13	00010011
20	00100000	14	00010100
21	00100001	15	00010101
22	00100010	16	00010110
23	00100011	17	00010111
24	00100100	18	00011000
25	00100101	19	00011001
26	00100110	1A	00011010
27	00100111	1B	00011011
28	00101000	1C	00011100
29	00101001	1D	00011101
30	00110000	1E	00011110
31	00110001	1F	00011111
32	00110010	20	00100000
33	00110011	21	00100001
34	00110100	22	00100010
35	00110101	23	00100011

********** END OF ARTICLE **********

ASSEMBLY LANGUAGE CLASS

The assembly language class starting at 10 am on meeting days, which has had falling numbers over the last few meetings, ceases from the March 1993 meeting.

AT THE 3RD APRIL 1993 MEETING...

The Model Train set controlled by a TI99/4A computer via the Wire I/O, will replace the Assembly class at the April 1993 meeting. The train set will be there for discussion on control systems using the Wire I/O from 10am. (This is especially for one member, but any other members who want to join in will be welcome.) After 1pm, the kids (both young and old) who just want to play trains will be welcome to do so. New additions to the train set since the Faire include a semaphore signal, extra trees and a model Stephenson's Rocket train.

********** END OF ARTICLE **********

WORD PROCESSING
PART 4

by Col Christensen
Brisbane User Group

Last month's article covered a number of dot commands which are placed in a text file and used as instructions by the Text Formatter. Some of these have numerical parameters following them. Reviewing one point, a dot command such as .LM8 has the absolute value of 8, and sets the left margin to the 8th print column whereas .LM+4 with a relative value of 4 sets the left margin at 4 columns greater than the previous one. All relative values in dot commands refer back to the previous setting of that command except for .IN+n which refers to the left margin's setting.

HIGHLIGHTING AND SPECIAL EFFECTS

Certain characters are used by the Text Formatter as flags to effect some print output features. The characters in question are the & for underlining, @ for overstriking and ^ for required space. (ampersand, at and circumflex).

REQUIRED SPACE with ^

The circumflex (^) tells the Text Formatter that words joined by it are to be treated as one word during underlining, overstriking, filling text or adjusting text. For example, you may want to be sure a group of words such as 24 January, 1992 will appear all on one line and not be split partly on one line and partly on the next. Just use the circumflex to replace each space like this, 24^January,^1992. The required space is useful also in underlining and overstriking groups of words as explained in the next two paragraphs.

UNDERLINING with &

Wherever the Text Formatter encounters the & symbol, the text from there to the next space character will be underlined. So if your text shows &IMPORTANT, the formatter will print IMPORTANT.

But if you wish to underline a group of words there are two ways to go about it. The first is to use the & symbol before each word. Typing &The &Storm &King will print to The Storm King.

The other way is to tie the words together with the required space. In &The^Storm^King, the whole title, spaces and all, will be underlined to become The Storm King as the printout. Although this is the preferred and easiest method, there can be a small price to pay in the FI;AD format. If the title is towards the end of a line and cannot be fitted as a unit on that line, it will be wrapped in full onto the next line. That would leave the previous line with a great number of spaces between words. You may not find the appearance of such a spacy line very elegant.

OVERSTRIKING with @

In a similar way to underlining in the paragraph above, the symbol @ is used to invoke overstriking. In doing this, the Text Formatter prints over the particular word four times before finally performing a line feed. The dark printing resulting stands out very clearly and is more prominent than even the printer's emphasized style of print. If a group of words is required to be overstruck, the required space symbol (^) can be utilised in a similar fashion to that shown in the paragraph on underlining.

Although the formatter removes the & and @ flags for underlining or overstriking, you can still make it print one of these characters. Just type the character twice and the formatter will "understand".

TRANSLITERATES

This unusual word (Latin: litera=a letter and trans=across) simply means the use of one character to represent either another character or even a group of characters. That's like assigning a "¦" as a kind of variable to represent some long word or some printer code that might, say, change the print mode to enlarged italic elite print.

The formatter, when it encounters a circumflex (^) or an asterisk anywhere in the text or a period (.)placed first on a line will remove it and execute some routine that is flagged by that symbol. To be able to purposely print one of those three, you need to transliterate some other little used characters to represent any you want to print. e.g.

```
.TL 124:94=,     "¦" becomes "^"
.TL 96:42=,      "`" becomes "*"
.TL 95:46=,      "_" becomes "."
```

The =, symbols above represent the carriage return character that appears on the screen when the ‹ENTER› key is pressed. The Formatter, on receiving the first character, will intercept it and send the second character to the printer instead.

```
.TL 123:70,85,78,78,69,76,87,69,66=,
```

In this one the character "{" is transliterated to a whole group of characters. Note the commas to separate each. So, in fact, wherever the formatter encounters a "{", it prints the word "FUNNELWEB" instead. Unfortunately, it does not format the rest of the printed line accordingly. In place of the one character, it prints 9 so there would be an additional 8 characters extending past the right margin with FI;AD in force.

```
.TL 125:27,88,1,28=,
```

The "}" character this time is interpreted to be the string of four characters following the colon in the TL. This string is a printer code for setting the left and right margins. It is probably shown in the printer manual as ESC "X" (n1) (n2). In the TL, the 27 is the ESC character, 88 is the "X", and the 1 and the 28 are the left and right margin settings for the printer. The printer, on receiving these four characters, will interpret them as a printer control code sequence and remove them from the input stream of characters. They will not be printed out,

but from here on, printing will only take place on lines that are 28 characters long. Because, in place of one character in the text (the "("), none were printed, the print line will be one character short of the right margin.

INCLUDE FILE

This is another dot command that adds versatility to the Formatter. The command could look like .IF DSK1.RESUME2. When the Text Formatter encounters such a command, it prints the contents of that disk file too. The command can be at the beginning, the end or in the middle of the main file. There is no limit to the number of IFs used in the main file but IFs cannot be chained, i.e. a file that has itself been IFfed cannot have an IF command in it. Suppose you have done up a review that occupies 4 disk files named REVIEW1 to REVIEW4. Then to format them what you can do is to create another file to print out the whole review such as:

```
.CO Review:- The Storm King⌐
.CO Date :- 23 March, 1991⌐
.LM10;RM70;IN+8;FI;AD
.HE^^^^^^^^^^^@THE^STORM^KING⌐
.FO^^^^^^^^^^^^^^^^Page^%⌐
.IF DSK1.REVIEW1⌐
.IF DSK1.REVIEW2⌐
.IF DSK1.REVIEW3⌐
.IF DSK1.REVIEW4⌐
```

A few more dot commands have cropped up here.

COMMENT

The .CO flags a comment just for the benefit of the reader. The whole line is ignored by the Formatter.

HEADER

.HE is the dot command that produces a heading on each page printed. The header is placed on the first line on the page and the text begins immediately on the next line. In the above example, the title, THE^STORM^KING would be printed at the top of each page. The eleven circumflexes forces the title to be spaced over 11 positions from the indent position and the @ ensures that the heading is overstruck four times. Reusing the command, .HE, without anything after it cancels any previous header command. If wording follows a new .HE, the new one will be printed.

FOOTER

.FO ensures that a footer is printed at the bottom of each page just after a blank line. The dot command can be followed by any text to be printed and/or the % character if required. The % instructs the Formatter to print consecutive page numbers. The example above will print both the word "Page" as well as its number. Likewise, page numbers can also be included in headers.

It seems that the Funnelweb formatter prints out three lines less than the .PL command setting. On a setting of 60 lines per page, for example, the format printed is:

```
1 line either header or blank.
2 blank lines
52 lines of text.
1 line blank.
1 line either footer or blank.
```

Concerning page numbers in headers or footers, what if we want to start numbering the pages from something other than one or even skip a page number somewhere where we intend to use a page for illustration purposes? Well, there is another command to take care of that.

PAGE NUMBER SET AND RESET

.PA followed by a number sets header or footer pages to begin from that number. Relative values such as +2 or -1 can also be used to reset page numbers.

********** END OF ARTICLE **********

JUST A LITTLE BIT HELPS

by Jim Peterson

I am not going to say that everybody should learn to program. Most folks can get along just fine by using the programs written by others.

But, every once in a while you need to do something that would be so easy for the computer to do, and so hard to do without a computer - but no one has written a program to do it. That is when just a little bit of programming knowledge can be a lifesaver!

In a previous article, I described my genealogy program, and the index for it I had created in Funnelweb by setting the tabs at 1 for the person's index number, 5 for the first name, 36 for the surname, 56 for the paragraph number, 61 for the father's index number and 66 for the mother's, and 71 for the spouse.

I have been busily adding names to that index until I am now up to well over 900, and I realize that I have goofed!

I set that second tab to allow space at left for a 1- to 3-digit index number and a space, but I will soon be going past 999. I need an extra space there. I allowed more than enough spaces for the names, so I can take some of those away. While I am at it, I would like to allow space for second and third spouse numbers at the right, and I want to keep the total spaces well short of 80 so I can print the index in two columns of elite condensed.

Am I going to retype 900+ lines, and undoubtedly make some errors while doing so? No way! Never do anything that you can make the computer do for you.

All this requires is knowing how to open one file to read from and another to write to, read a record from the one file, manipulate it a little, and write it to the other file.

The file part is extremely simple, just OPEN #1:"DSK1.OLDFILE",INPUT and OPEN #2:"DSK2.NEWFILE", OUTPUT - or whatever drives and filenames I choose. The computer takes it for granted that I am using DV80 files, so I do not have to tell it. Even the INPUT and OUTPUT could be omitted, but it is best not to. If you do not tell it otherwise, the computer will allow you to accidentally write to the file you meant to read from, and that is disastrous!

The reading and writing is equally simple-

```
110 LINPUT #1:M$ :: PRINT#2:M$ :: IF EOF(1)<>1 THEN
110 ELSE CLOSE #1 :: CLOSE #2
```

Always use LINPUT when reading a text file, because INPUT will only read the record up to the first comma it finds.

EOF is set to zero until the last record in the file has been read, when it is set to 1, so IF EOF(1)<>1 means "if the end of the file opened as #1 has not been reached" - in which case, go back to the same line number and read another record, but if you have read everything, close the files - and especially close the one you have written to, or you will have wasted your time.

But, I wanted to change the lengths of the fields in each record before I move it to the new file. For that, I need to know only how to use two commands - SEG$ and &.

SEG$ extracts a part of a string - a string, in this case, is the record I get with that LINPUT, which is one line of up to 80 characters that I keyed in with Funnelweb. SEG$(M$,1,4) will give me 4 characters starting with the first character of M$; SEG$(M$,5,25) will give me 25 characters starting with the 5th. And & will put those two together into one string. I want to keep that first 4-character field but expand it to 5 characters, so SEG$(M$,1,4)&" " . The next two fields, starting at 5 and 36 and consisting of 31 and 20 characters, I want to cut to 20 and 16 characters, so - SEG$(M$,1,4)&" "&SEG$(M$,5,20)&S EG$(M$,36,16). And I go on to add a space to each of the next three fields, and the whole program looks like this -

```
100 OPEN #1:"DSK1.INDEX1",INPUT :: OPEN #2:"DSK1.NEWFILE
",OUTPUT
110 LINPUT #1:M$ :: PRINT #2:SEG$(M$,1,4)&" "&SEG$(M$,5,
20)&SEG$(M$,36,16)&SEG$(M$,56,5)&" "&SEG$(M$,61,5)&" "&S
EG$(M$,66,5)&" "&SEG$(M$,71,3)
120 IF EOF(1)<>1 THEN 110 ELSE CLOSE #1 :: CLOSE #2
```

And in a few minutes, that program does the job, saving many hours of work!

********** END OF ARTICLE **********

DV80 TO PROGRAM CONVERSION

by Jim Peterson

John "Jeb" Hamilton of the Central Iowa User Group was the first to realize, several years ago, that a DV80 listing of a Basic or Extended/Basic program could be converted to a DV163 file and then merged in and run as a program. I no longer have his program in my library, but this is a quick and dirty version of it-

```
100 DISPLAY AT(12,1)ERASE ALL:"Input file? DSK":"":"Outp
    ut file? DSK"
110 ACCEPT AT(12,16):A$ :: ACCEPT AT(14,17):B$
120 OPEN #1:"DSK"&A$,INPUT :: OPEN #2:"DSK"&B$,VARIABLE
    163,OUTPUT :: LINPUT #1:M$
130 LINPUT #1:M$ :: IF LEN(M$)>78 AND EOF(1)<>1 THEN LIN
    PUT #1:M2$ :: M$=M$&M2$
140 X=POS(M$," ",1):: Y=VAL(SEG$(M$,1,X-1))
150 PRINT #2:CHR$(INT(Y/256))&CHR$(Y-256*INT(Y/256))&"!"
    &SEG$(M$,X+1,255)&CHR$(0)
160 IF EOF(1)<>1 THEN 130 ELSE CLOSE #1 :: PRINT #2:CHR$
    (255)&CHR$(255):: CLOSE #2
```

To try that out, key in this useless little program-

```
100 CALL CLEAR
110 FOR J=1 TO 10
120 PRINT J
130 NEXT J
140 END
```

List that to disk by LIST "DSK1.80". Then run the above converter program, answer the input prompt with 1.80 and the output prompt with 1.163. After it runs, enter NEW, then MERGE DSK1.163 and then LIST. This is what you should see

```
100 !CALL CLEAR
110 !FOR J=1 TO 10
120 !PRINT J
130 !NEXT J
140 !END
```

Type 100 and FCTN X to bring line 100 to the screen with the cursor on the "!". Type FCTN 1 to delete the "!" and repeat with FCTN X and FCTN 1 to delete all the others. Then enter RUN and it should do so!

All that the program does is delete the blank first line of the listing, convert each program line number to tokenized format, add a CHR$(0) end-of- line marker to each line, move the record to a DV163 file, and add the double CHR$(255) end-of-file marker.

The result is a merge format program composed of REM statements; when you delete the "!" REM indicator, these become program lines.

There is just one problem. A LISTed program is a DV80 file, consisting of records of 80 characters or less, but a program line in Extended/Basic can be keyed in up to 140 characters long, and can be forced even longer (as I often do!) When such a line is LISTED, it is broken into 80-character records, which confuses the conversion program completely.

Line 130 of the conversion program attempts to resolve that problem. If a record is more than 78 characters long (because it could have been an 80-character line ending in a blank, which would become a 79-character record without the blank) it is taken to be most probably the first part of a long program line; another record is read in and tacked onto it.

However, this creates another problem, as you will find if you LIST the converter program and then try to convert it back to a program - line 140 will be tacked onto line 130 because line 130 is 79 characters long.

The best fix for this is to load the DV80 file into Funlweb and print out a hard copy; use a ruler to draw a vertical line after the 78th characters; mark any program line that ends on the 79th or 80th characters, delete those characters, save the listing, run it through the converter, merge it in and key those deleted characters back in - still much easier than keying in an entire listing.

After John Hamilton published his discovery, several authors wrote their own versions. It was suggested that programs could be written in text format, using the superior editing features of TI-Writer or Editor Assembler, and then converted to program format. Personally I was satisfied with the editing features of Basic and was not about to give up its syntax error-catching capability, so I never tried this method.

However, nowadays several hundred text files of newsletter articles are available on the Clearing House BBS, and other newsletters are being circulated on disk. Many of these articles contain program listings, and it would be much easier to extract and convert them than to print them out and key them in.

Later on, John Ford wrote a more complex converter called XLATE, which eliminates the need to delete all the "!" by converting the ASCII text file directly to tokenized merge format. It also checks for syntax errors and corrects them or reports them on-screen . If the LISTed program had regularly sequenced line numbers, it will check these to see whether records should be combined, which should greatly improve accuracy - I have not tested it enough to say how foolproof it is.

Blanks at the end of a DV record are dropped, so if the 80th character of a long program line is a blank, when the line is broken into two records and then recombined the blank will be missing. For instance, if the blank between FOR and J in FOR J=1 TO 10 happens to be the 80th character, it will recombine as FORJ=1 TO 10. This results in a SYNTAX ERROR referencing the line number, which is therefore easy to spot and correct. The same problem can cause the string "John Doe" to become "JohnDoe".

The above conversion programs are intended for listings in 80-column format. However, many of the listings within text articles have been reformatted to 28-column or 40-column width, or listed in those widths with Triton Super Extended/Basic.

Fortunately, there is an alternative. Curtis Alan Provance has written a truly remarkable program in assembly, called TEXTLOADER, which will convert a DV80 file directly into a program in memory, and will handle the shorter line lengths, although with increased chance of error because the method of detecting new lines is far from foolproof.

I have not tested this program extensively, but have found only two major problems. The one is with records ending in a dropped blank, as described above; these are easy to correct. The other is with split referenced line numbers. For instance, if a line ends in GOSUB 120 :: GOTO 200 and splitting of records turns this into GOSUB 1 and 20 GOTO 200, you will find the line ending with GOSUB 1 and a new line 20 :: GOTO 200 at the beginning of the program. Comparison with the original listing makes this easy to correct.

TEXTLOADER loads into memory and remains there, so that you can load other text files by simply typing - CALL LINK("OLD"&"DSKn.filename"). The file loads and converts rapidly, displaying each line as it does so. Sometimes a line which has been corrupted will be reported as a syntax error and omitted, but sometimes it will be omitted without being reported, and sometimes it will not be detected until you try to run the program. Occasionally, especially when working with 28-character lines, you may get all sorts of invalid error messages. Apparently the program in memory differs from the screen display, and it may be impossible to debug in such cases.

Other features allow you to merge a converted text file into a programm in memory rather than overwriting it, and to read and run a batch file of command type instructions, such as -

```
CALL FILES(1)
NEW
RUN "DSKn.bigprogram"
```

An improved general-purpose memory image program loader is also included.

XLATE is a public domain program, available on my TI-PD disk #1083. TEXTLOADER is a fairware program available on my TI-PD disk #1104.

********** END OF ARTICLE **********

RELISTING PROGRAMS

by Jim Peterson

At the last meeting, our editor asked me about ways to convert listed programs to 28-column width, and to convert listed programs to runnable programs. A couple of days later, I had a phone call from a user asking about the same thing. And, I have received a few newsletters with reprints of an article describing a method of listing to the printer in 28-column format.

Why list in 28-column format? Because that is the way a program appears on the screen. It is much, much easier to key in a program accurately when it is published in 28-column format, because you can edit your work by checking the position of characters in relation to the line above - especially when the program contains long stretches of blanks, or long hex codes.

About that method currently being reprinted - it does not work. At least, it does not work properly with Extended Basic programs. The idea is that you open the printer and send it ASCII codes 27 81 28, which sets the right margin at 28. You can get the same result by OPEN #1:"PIO",VARIABLE 28 .

The problem is that Extended Basic program lines can be keyed in up to 140 characters long, and can be forced considerably longer. When you LIST a program to disk, it is saved in DV/80 format. Any line longer than 80 characters is broken into separate 80 character records. When you break those records into 28-character segments, you have program lines stopping in the middle and then continuing on the next line. They can still be keyed in correctly, if you realize what has happened, but the listing will not be in screen format, which is the whole purpose of using 28 columns.

Besides, you probably do not want to output to the printer. You want to output to disk, so you can incorporate the listing into a text article, as I am about to do. So, what to do? If you have the Triton Super Extended Basic module, it is as easy as pie. Just - LIST "DSK1.LISTING":28:1-32766. It will do a perfect job but the listing will be in DV/28 format, which will not load into Funnelweb. So I will now write a little program, save it, list it with my Super Extended Basic, and then load my little program to convert the DV/28 file into a DV/80 file which I will insert right here -

```
100 DISPLAY AT(10,1)ERASE ALL:"Input file? DSK":"":"Outp
ut file? DSK" :: ACCEPT AT(10,16):IN$ :: ACCEPT AT(12,17
):OUT$
110 OPEN #1:"DSK"&IN$,VARIABLE 28,INPUT :: OPEN #2:"DSK"
&OUT$,OUTPUT
120 LINPUT #1:M$ :: PRINT #2:M$ :: IF EOF(1)<>1 THEN 120
 ELSE CLOSE #1 :: CLOSE #2
```

But you do not have the Triton module? Well, several years ago I wrote a 28 column converter which will do the job perfectly. It will also optionally replace and transliterate those characters that get messed up when you print a program listing through the Formatter. It will even recognize unprintable blank characters which have been keyed in with the CTRL key and print their key letter underlined. That program was published in Tips From The Tigercub #18 with an upgrade in #21. It is available on my TI-PD disk #1015.

That program does require that the listing have standard line number spacing, numbered by tens from 100. If you are starting with a listing which is not in that format, this one will do the job but not as easily, because you have to first insert a carriage return at the end of each program line. To do that, load the listing into the Funnelweb Editor, press CTRL O to get the hollow cursor and CTRL U to get the underline cursor, go to the end of each program line with the arrow keys and press M.

```
100 DISPLAY AT(3,6)ERASE ALL:"PROGRAM RELISTER":"":" Wi
ll reformat a LISTed XBasic program from any linelength
to any other length."
110 DISPLAY AT(8,1):" Each program line (not file line)
must end in a carriage return."
120 DISPLAY AT(12,1):"Input filename?":"DSK" :: ACCEPT A
T(13,4):IF$ :: DISPLAY AT(15,1):"Output filename?":"DSK"
 :: ACCEPT AT(16,4):OF$
130 DISPLAY AT(18,1):"Present line length?" :: ACCEPT AT
(18,22)SIZE(2)VALIDATE(DIGIT):A
140 DISPLAY AT(20,1):"Reformat to what length?" :: ACCEP
T AT(20,26)SIZE(2)VALIDATE(DIGIT):X :: IF X=A THEN 130
150 OPEN #1:"DSK"&IF$,INPUT :: OPEN #2:"DSK"&OF$,OUTPUT
:: IF X<A THEN 230
160 IF EOF(1)THEN 270 :: LINPUT #1:M$ :: L=LEN(M$):: IF
POS(M$,CHR$(13),1)=0 THEN 180
170 IF P+L<X+1 THEN PRINT #2:M$ :: P=0 :: GOTO 160 ELSE
PRINT #2:SEG$(M$,1,X-P)&CHR$(13):SEG$(M$,X-P+1,255):: P=
0 :: GOTO 160
180 IF L<A THEN M$=M$&RPT$("",A-L):: L=A
190 IF P=0 THEN PRINT #2:M$;:: P=L :: GOTO 160
200 IF P+L<X THEN PRINT #2:M$;:: P=P+L :: GOTO 160
210 IF P+L=X THEN PRINT #2:M$&CHR$(13):: P=0 :: GOTO 160
220 PRINT #2:SEG$(M$,1,X-P)&CHR$(13):SEG$(M$,X-P+1,255);
:: P=LEN(SEG$(M$,X-P+1,255)) :: GOTO 160
230 IF EOF(1)THEN 270 :: LINPUT #1:M$
240 L=LEN(M$):: IF L+P>X THEN PRINT #2:SEG$(M$,1,X-P)&CH
R$(13):: M$=SEG$(M$,X-P+1,255):: P=0 :: GOTO 240
250 IF M$=CHR$(13)THEN 230
260 IF POS(M$,CHR$(13),1)<>0 THEN PRINT #2:M$ :: P=0 ::
GOTO 230 ELSE PRINT #2:M$;:: P=LEN(M$):: GOTO 230
270 CLOSE #1 :: CLOSE #2
```

That one is also on TI-PD 1015.

Now, about converting listings to programs, without having to key them in - well, let's save that for next month.

********** END OF ARTICLE **********

CHART 1

GIF-MANIA / TI ARTIST CONVERSION MULTI TI 40-COLUMN SCREEN MAP

A	B	O	P	C.
C	D	Q	R	D.
E	F	S	T	E.
G	H	U	V	F.
I	J	W	X	G.
K	L	Y	Z	H.
M	N	A.	B.	I.

CHART 2

PAGEPRO SHEET AREA
SCREEN PLACEMENT MAP

A	B
C	D
E	F
G	H
I	J

TABLE 2

PAGEPRO SHEET AREA
SCREEN PLACEMENT CO-ORDINATES

AREA	LINE	COL
A	1	1
B	1	33
C	17	1
D	17	33
E	33	1
F	33	33
G	49	1
H	49	33
I	65	1
J	65	33
AREA	VER	HOR

TABLE 1

GIF-MANIA TI ARTIST CONVERSION
ZOOM MODE SCREEN CO-ORDINATES

SCREEN	HOR	VER
A	0	0
B	256	0
C	0	192
D	256	192
E	0	384
F	256	384
G	0	576
H	256	576
I	0	768
J	256	768
K	0	960
L	256	960
M	0	1152
N	256	1152
O	512	0
P	768	0
Q	512	192
R	768	192
S	512	384
T	768	384
U	512	576
V	768	576
W	512	768
X	768	768
Y	512	960
Z	768	960
A.	512	1152
B.	768	1152
C.	1024	0
D.	1024	192
E.	1024	384
F.	1024	576
G.	1024	768
H.	1024	960
I.	1024	1152

GIF TO PAGE PRO CONVERSIONS

by Alf Ruggeri

GIF MANIA MAY BE USED IN TWO WAYS:

A. As a viewer of high resolution images created on PC's.

B. As a converter of these images into monochrome TI-ARTIST graphics for ultimate conversion as images for desk top publication via PAGE PRO.

GIF MANIA AS A VIEWER

I can still remember the gasps of admiration that GIF Mania drew, from a less than impressionable member of my family, on first observing from the doorway of my study, images such as DREAM or CITY, materialize in a holland blind fashion, as that family member came to hail me to the realities of mealtimes and forgotten chores.

I must admit I was quite impressed with some of the images at the time, as an inexpensive colour TV set fed via a VHF modulator served as a monitor. Sometime later a proper monitor was acquired, certainly not for the sole purpose of improved GIF file inspection but with an anticipated video linearity improvement.

The improvement in video reproduction certainly resulted but alas and alack it in no way improved GIF viewing, rather quite the reverse. Previous charming images like DREAM and CITY now appear as the end product of a four year old toddler's crayon battle with a colouring book. The outlines of the image's components no longer serve to contain all the pigmentation previously allocated to a particular region. The result could not be described as pleasant.

Of course the same GIF files when observed through Achim Liese's GIF viewer via a MECHATRONIC 80 column card are certainly spectacular.

In all fairness I am sure that the colour resolution problem is not a shortcoming of GIF Mania but rather the limitations of our 256 x 192 pixel with 16 colour capability screen format.

GIF MANIA AS A CONVERTER

GIF Mania is an excellent utility in the conversion of GIF images to TI-ARTIST format. GIF images as large as 1264 pixels in width and 1242 pixels in height can readily be converted to TI99/4A graphics via a relatively simple procedure.

A colour to monochrome threshold control, adjustable through 255 steps, enables subtle black / white contrast in the conversion process.

When used as a converter with my process, it is necessary to have access to Pix Pro, PICASSO, PAGE PRO.

The Achim Liese program mentioned above also provides conversion to TI99/4A graphics and Larry Saunders, on whose system I've seen that program demonstrated, may find time in his busy schedule to elaborate on its features, when the 80 column population in our community increases via the TIM board.

A FITTING TASK FOR THE CONVERTER

In the process of deciding the occasion theme for my Greeting Card demonstration / production at the TI-Faire last November, I came to some conclusions:

1. The TI-Faire presentation date's proximity to the end of the year logically suggested a Christmas theme.

2. I did not have a sufficient number and variety of Christmas motif graphics that I thought represented the range of imagery that conveyed Christmas greetings.

3. The only way that I was going to obtain the graphics that I needed was to scan them on a PC with subsequent conversion to the world of the TI99/4A through GIF Mania.

SOURCE OF IMAGES

The pictures I chose originated from sources as varied as my children's old school magazines, well used bedtime story books, some of my gardening books and magazines.

Although I found it very tempting to use such icons of Australiana as the Gum Nut characters from May Gibb's books, permission to use them, even if only for a limited promotional activity such as the TI-Faire, was explicitly refused by the copyright authorities in Melbourne. Yes! I actually wrote to them.

SCANNING THE PICTURES

The pictures were scanned in on my brother-in-law's 286 PC via the use of a hand held roller scanner using the Dr.Genius software. The scanned files were saved as .PCX format and further processed as GIF to reduce file size and enable X-MODEM protocol transmission.

PC GIF TO TI99/4A GIF TRANSFER

The scanned GIF files could have been transferred by the normal telephone / modem method I described in my initial review of GIF Mania, but even at 1200 bauds, I would have been straining the bounds of familial tolerance with the phone engaged for an interminable length of time.

Instead I decided to try Rolf Schreiber's NULL MODEM method. The 286 PC and the TI99/4A were physically located side by side and the respective RS232 ports were interconnected by a standard null modem cable. The 286 PC was set up in the NETCOM X-MODEM file upload format and the TI99/4A set up in the TELCO X-MODEM file download. After a few frantic inspired incantations and a quick phone call to Geoff Trott, my brother-in-law and I were able to send files from the PC to the TI.

Data transfer at 9600 bauds was possible but with too many errors, probably due to the null modem cable length. The error rate at 4800 bauds was found tolerable and the entire PC to TI transfer took place without any further real traumas.

CONVERSION PROCEDURE OVERVIEW

There are two modes in which GIF Mania will display a GIF image. They are controlled by the user response to the "Condense? (Y/N)" prompt.

A "Y" response will try to display the entire image onto the screen. Unless the image is small to begin with, the resulting screen presentation will be vertically and horizontally compressed. With very large images (spatial, not file size) the screen result will probably be unrecognisable.

A shortcoming of this feature is that in spite of the anticipated image compression, a large GIF will not be proportionally displayed. The full compressed width of 1264 pixels is presented, but only 31% i.e. 384, of the full 1242 pixel height. The expected height, derived from the TI screen dimensions ratio, is 192 : 256 = 75% of 1242 i.e. 932 pixels, does not eventuate. This observation may seem trivial, as as indicated in the previous paragraph, the resultant display may be meaningless. However 75% height display would at least serve to verify image occupation below the 384th pixel row. The shortcoming became evident when I could not locate image areas that I knew had been successfully scanned in entirety and saved away as files.

An "N" response will allow a viewing window 256 pixels wide and 192 pixels high, the standard TI99/4A pixel screen area, to be located anywhere on a GIF image area. The numerical values entered for the "Left Shift:" (horizontal co-ordinate) and "Up Shift:" (vertical co-ordinate) prompts will determine the location of the window's top left hand side corner. The default for these two prompts are 0 and 0.

With the benefit of the movable uncondensed window feature, the concept of the conversion process can best be understood through a simple scrapbook analogy.

Consider the GIF image area (it can be as large as 1264 pixels wide and 1242 pixels in height) as a newspaper or magazine page that is going to be clipped out in even size portions. The newspaper or magazine clippings (256 pixel width by 192 pixel height) are then reassembled in correct adjacent order and pasted onto a scrapbook page (a PAGE PRO page).

THE CONVERSION PROCESS

1. THRESHOLD DETERMINATION

If a coloured GIF image requires conversion choose "M" from the "Color Select (I/D/G/M):" option. A brightness figure (threshold control) adjustable through 255 steps is presented. The limits are 1 for very dark and 255 for very light.

Note: the threshold control has little effect on predominantly dark portions of coloured images and regrettably no effect at all on black and white images.

I regret the ineffective threshold control with black and white GIF's because I had found the feature quite useful with coloured GIF's and wrongly assumed that it would function just as well with all type of GIF's. This is in view of the fact that I virtually disregarded a black to white tonal balance adjustment feature of the PC scanning program that I used.

If I might be allowed a little a whimsical digression for the sake of levity as a relief from this technical discourse. I would like to question the wisdom of software designers in assigning the jargon name of "DITHERING" to the above PC tonal balance adjustment. A rather strange choice of word, is it meant to be synonymous with indecision or vacillation, or is it an inadvertent statement identifying our social interface with the computer age, for simply filling in time between periods of eating and sleeping. This person has not missed too many meals through his affliction of computeritis, mainly due to the vigilant surveillance of the forementioned family member, and the fact that pangs of hunger limit the inclination to tap a few more lines of code or text onto a keyboard. But as to falling asleep in front of a computer, whilst waiting for a certain loop to run its merry course, well!, I must admit I have sometimes wondered, why the monitor screen has gone blank, when only what seemed just a little earlier, the loop had only just commenced.

Well! back to the process.

2. TI99/4A WINDOW POSITIONING

Following the scrapbook analogy of the GIF image (newspaper page) being clipped out as adjacent portions or areas of 256 hor. x 192 ver. pixels, the procedure is facilitated by the use of my CHART 1 and TABLE 1.

CHART 1 is a comparison of pixel occupation between the non 80 column TI99/4A screen and that of the PC SUPER VGA. Approximately a ratio of 1 to 35.

The TI screen area names are identified by one of 35 alphabetical symbols. These 35 symbols are cross referenced in TABLE 1 as horizontal and vertical co-ordinates for the top left hand corner of each of the 35 areas.

3. GIF TO TI-ARTIST CONVERSION

After each of the 256 x 192 areas have been displayed:

a/ Select the "Save TI Artist" option.
b/ Type in a path and filename for the converted image.

The filename must not be longer than 8 characters as GIF Mania will attach the TI-ARTIST file extension. To maintain conformity to CHART 1 and TABLE 1, the filename need only be the area alphabetical identification symbol.

The conversion process generates a _P (picture) and a _C (colour) TI-ARTIST files.

In the conversion procedure that I use, the _C file is superfluous and can be discarded immediately after the conversion to TI-ARTIST has taken place. However in the interest of procedural continuity, rather than:

i/ Halt GIF Mania after each conversion.
ii/ Delete the generated _C file.
iii/ Reboot GIF Mania.
iv/ Retype the GIF file name and prompt responses.

It is far more convenient to determine the maximum number of _P and _C files (both 25 sectors each) that can be accomodated on the floppy or RAM disk that is going to be used as storage. The number is divided by two and used as a conversions limit for that particular disk. When the limit has been reached, the superfluous _C files can be deleted.

4. TI-ARTIST TO PAGE PRO CONVERSION

Run Pix Pro to convert the 256 x 192 pixel TI-ARTIST areas to PAGE PRO images using the appropriate option.

As per the GIF to TI-ARTIST conversion process, the PAGE PRO converted filenames can be the area alphabetical identification symbols.

5. PAGE PRO IMAGES ASSEMBLY

The PAGE PRO images representing segments of the original GIF image areas are reassembled in correct adjacent order onto a PAGE PRO page (scrapbook page; viz scrapbook analogy) by the use of CHART 2 and TABLE 2 as map and placement co-ordinates.

Users familiar with PAGE PRO, will realize that the pixel area of a PAGE PRO page is 480 pixels wide and 792 pixels in height. This area therefore does not not even represent two full sets of five side by side converted segments. However it is the most efficient use of the "paste up" sheet facility of PAGE PRO. The image areas seemingly lost by the overhang at the right hand side page edge and the bottom of the page can be recovered in succeeding page "paste up's".

Although it should be immediately obvious, that the PAGE PRO sheet screen placement map of CHART 2 and the associated co-ordinates of TABLE 2, are for utilizing the sheet as a utility "paste up" platform, for any frame of two by five adjoining GIF image areas, I realize that the similarity of the area identification symbols with those of CHART 1 and TABLE 1 may be confusing. It is sufficient to say that the symbols of CHART 2 and TABLE 2 only serve to ensure sheet correct adjacent assembly. The images that will occupy these areas are the converted segments from CHART 1. If confusion still prevails, please read all of step 5 again.

6. PAGE PRO TEST PRINTOUT

The assembled composite PAGE PRO page, or pages, can be printed out to allow inspection of up to ten adjacent segments to determine what part of the page can be used as picture for a particular application, and whether that part will require graphic clean-up in PICASSO.

PICASSO is preferred to TI-ARTIST as a graphic clean-up utility because:

a/ Each pixel be it on or off is identified as a hollow or full square in a grid structure whilst in the ZOOM mode.
b/ The PICASSO DV80 file (85 sectors) is 480 pixels wide and 336 pixels in height compared with the 256 x 192 pixels of TI-ARTIST. The additional area reduces the time involved in clean-up of adjacent assembled segments.

7. PAGE PRO TO PICASSO CONVERSION

The PAGE PRO area requiring clean-up is saved away as a NEWPICTURE (FCTN 7 - see PAGE PRO documentation). Pix Pro is run and the NEWPICTURE is converted to a DV80 PICASSO file.

8. GRAPHIC CLEAN-UP IN PICASSO

a/ PICASSO is run.
b/ Press the "G" key and type in the path and filename of the PICASSO file.
c/ Toggle the appropriate pixels to clean up the graphic.
d/ Press the "S" key and type in the path and filename of the cleaned up PICASSO file, the filename can be the same one as per step b/.
e/ Convert the PICASSO file to PAGE PRO via Pix Pro.
f/ The graphic can be trimmed to whatever is the required size by loading the PAGE PRO image into a PAGE PRO page and saving away as a NEWPICTURE.

9. PAGE PRO GRAPHIC MANIPULATION

The cleaned up PAGE PRO graphic can of course be further enhanced by:

a/ Flipping
b/ Rotation
c/ Ghosting
d/ FX-ing (scaling)

via the PAGE PRO UTILITIES to produce the desired effect.

SCANNING RATES AND PRINTOUT SIZE

I chose the lowest scanning available rate i.e. 100 D.P.I. on my brother-in-law's PC, knowing that the PAGE PRO printer drive file would produce a maximum of 60 D.P.I. on my EPSON printer i.e. 480 dots across a page width of 8 inches (the specifications are of US origin - thus not metric).

The result of using the 100 D.P.I. scanning rate produced a magnification factor of 1.67 of the image before scan and the resulting PAGE PRO hardcopy. A scanning rate of 200 D.P.I. was tried and predictably produced a magnification of 3.3.

OVERALL GIF TO PAGE PRO CONVERSION TIME

There is no denying that this conversion procedure is slow. The slowest part is in step 3 whilst GIF Mania is searching for the lower 256 x192 pixel areas. The longest search time is as expected, for the bottom row of areas (if the GIF image is that tall) whereat the monitor screen turns black for about six minutes before the chosen area is displayed. Definitely a period for silent meditation, or reading a page or two of a book.

The rest of the procedure is one of frantic activity. However if it is followed as indicated, the result is well worth the effort.

As I mentioned at the beginning of this article, I know of only one other GIF viewer i.e. Achim Liese's program for an 80 column card configuration. If anybody is aware of another non 80 column GIF viewer - converter please let me know, I would be most interested.

A PLANNED FULL DAY TUTORIAL

After the TI-Faire last November, I decided that I would present a detailed tutorial of my conversion experience with GIF Mania at the May 1993 full day tutorial.

The planning was good. It allowed me to negotiate six weeks of neglected domestic chores as a result of preparation for the TI-Faire. Several work related field trips. Two groups of European relatives visiting Australia over a period of four months, with myself, invariably seconded by my parents as a tour guide. Preparations for my son's 18th birthday party.

Everything was going to plan until Ross Mudie phoned me very early in the new year. He had just successfully designed various routines that enabled GIF file download from Texpac, a feature that I had suggested just after my initial GIF Mania review. I downloaded several GIF files that Ross had placed on Texpac and sure enough the results were spot on.

Thanks very much Ross for your tremendous effort.

At Ross's request I mailed him 5400 sectors of GIF files, my entire collection, so that he could process them during his holidays, for future Texpac download.

Because of the GIF file Texpac download facility, I felt it was necessary to promote GIF Mania usage and possible renewed use of our BBS, by bringing my presentation forward, at least in text form. The result is this article.

If you are interested in GIF file conversion and making use of the excellent effort that Ross puts into maintaining our BBS, try out my procedure for yourself.

If you have any questions about my procedure, or GIF files in general, please come along to the full day tutorial and hopefully learn the answers.

Best regards
Alf Ruggeri

********** END OF ARTICLE **********

LEARN TO KNOW YOUR TI
LESSON 3
with Percy Harrison

This months lesson will be devoted to the use of the CALL CLEAR and LIST functions and finish up with a small program which will draw a picture of a very basic old fashioned car using characters of the alphabet.

LESSON 3 CALL CLEAR, LIST

Enter: NEW

Start each lesson with NEW to erase the memory and clear the screen.

PUT A PROGRAM INTO MEMORY

Now enter:
```
10 REM HOUSE
20 PRINT "LISTEN"
30 CALL SOUND(200,800,10)
40 CALL SOUND(300,600,10)
50 PRINT "DID YOU HEAR THE DOORBELL?"
```

Run this program.

ERASING THE SCREEN

Now enter: CALL CLEAR

CALL CLEAR is a command to erase the screen. It does not erase the program in memory. CLEAR means the same as "erase."

IS THE PROGRAM LOST FOREVER?

You can no longer see the program on the screen, but the program is not lost. The computer has stored the program in memory and we can ask the computer to show us the program again.

LISTING THE PROGRAM

Enter: LIST

and the computer will list the whole program on the screen. To see just one line of the program, ask for it by line number:

```
LIST 30
```

shows line 30 of the program.

THE MEMORY

The computer's memory is like a shelf of boxes.

The name of the box goes on the front of each box.

At the start, all boxes are empty and no box has a name.

When you entered:

```
10 REM HOUSE
```

the computer took the first empty box and wrote the name "Line 10" on the label. Then it put the command REM HOUSE into the box and put the box back on the shelf.

When you entered:

```
20 PRINT "LISTEN"
```

the computer took the second box and wrote "Line 20" on its label, then it put the statement PRINT "LISTEN" into that box and put it back into its place on the shelf.

What did the computer do when you entered line 30?

ERASING A LINE FROM MEMORY

To erase one line of the program, enter the line number with nothing after it.

To erase line 20

Enter: 20

You still see the line on the screen but do a LIST and you will see that line 20 is gone from memory.

When you enter just a line number with nothing after it, the computer:

finds the box with that line number on it

empties the box

and erases the name off the front of the box.

What does the computer do to the boxes when you give it the command NEW?

ADDING A LINE

You can add a new line anywhere in the program, even between two old lines providing the line numbers of the two old lines are not consecutive numbers. Just pick a line number between any two existing line numbers and type your new line in. The computer will put the line in its correct place.

Enter: NEW

Enter:
```
10 REM MORE AND MORE
20 PRINT
40 PRINT "MORE LINES WANTED"
```

List and run it. Now add this line:

```
15 PRINT "STILL"
```

List and run it again. Explain what happened.

FIXING A LINE

If a line is wrong, just type it over again.

For example, to change line number 40 in the above program:

Enter: 40 PRINT "NEEDS FIXING"

What did the computer do to the box named "Line 40" when you entered the line?

THE REM COMMAND

Use a REM command to put a title on your program.

Enter: NEW

```
10 REM REMARKS
20 CALL CLEAR
30 PRINT "LINE 10 DOES NOTHING
35 REM THIS LINE DOES NOTHING
RUN
```

What happens in each line of the program?
(Write your answers in the blank spaces)

Line 10.....

Line 20.....

Line 30.....

Line 35.....

REM means "remark" or "reminder."

Use REM to give a title to the program.

Use REM to write little notes in the program:

the notes are for you when you read the program

the notes are also for other readers.

Make the notes explain how the program works.

PICTURE DRAWING

You can use the PRINT command to draw pictures.

Here is a picture of an old model car. Enter NEW then enter this program.

```
10 REM STANLEY STEAMER
15 CALL CLEAR
20 PRINT
30 PRINT" XXXXXX"
40 PRINT"XXXXXXXXXXXX"
50 PRINT" O        O"
```

Do not forget to put the spaces in the PRINT lines! They are part of the drawing. Now run the program.

ASSIGNMENT 3:

1. Add a line to the STANLEY STEAMER program to make the car honk its horn.
2. What command will list line 10 of the program?
3. How do you tell the computer to list the whole program on the screen?
4. What does the computer do (if anything) when it sees the REM command?
5. What is the REM command used for?
6. Use CALL CLEAR, CALL SOUND, REM and PRINT to draw 3 flying birds on the screen. Make each bird peep.

I hope that you are going back over the previous lessons during the month so that you will become very conversant with the functions that each of them cover. Also you should try exercises of your own. A good start would be to study the TI book "User's Reference Guide" which came with your computer.

ANSWERS TO LESSON 2

Assignment Question 2-1

```
10 REM NAMES
20 PRINT "IMA"
30 PRINT "MARY"
40 PRINT "TAYLOR"
```

Assignment Question 2-2 and 2-3

```
10 REM NAMES
20 CALL SOUND(300,400,10)
22 CALL SCREEN(5)
25 PRINT " IMA"
30 CALL SOUND(300,600,10)
32 CALL SCREEN(10)
35 PRINT " MARY"
40 CALL SOUND(500,800,10)
42 CALL SCREEN(15)
45 PRINT " TAYLOR"
```

********** END OF ARTICLE **********

EXTENDED BASIC WEEKEND

TUTORIAL 17th. 18th.

APRIL

TECHO TIME

by Geoff Trott

Letter from Pierre Garoche

Firstly, all the best in 1993. Each month I am filled with admiration as I read the TND. I am struck by the diversity of the subjects. Every article published has useful information or valid opinions on the broached subjects. I often ask myself why the TND is so much better than other publications.
Perhaps all the members live far from each other and the BBS is an expensive link.
Perhaps the ratio of members with computer expertise is higher than normal.
Perhaps the authors do not expect to make money when they write an article.
Perhaps when a re-print of an article is chosen it is done with care and not just the most convenient at hand.
Perhaps when they commit themselves to do something it is done within the time limit.
I am sure that it is all this and more which explains the quality of the TND.

In the December 1992 issue, Ross Mudie asks around for subjects for beginners. This is a good idea. After I learned BASIC programming, I was interested in communicating via the telephone network. Is this a good subject for beginners with no PEBox?

I think that it may be interesting to have information about the system configuration of any TIsHUG member. Precise information about each device; manufacturer, type etc. should be collected. Perhaps a check-list could be published in the TND to help gathering the data and interpreting it.

Best regards, Pierre Garoche

Pierre may live a long way from Sydney but he is a great member of the club in that he is always doing something with and thinking about his computer and then writing to tell us about his thoughts. He is also quite knowledgable about assembly language and programming the hardware of his computer. The fact that he says nice things about the TND is neither here nor there (!?). I have done a bit of editing of his original letter to make it more readable without changing the meaning (I hope). The fact that it appears in Techo Time is because it was given to me at the last meeting and it does have a bit of technical content. In fact, I will now proceed to tell you the details of the system that I use to do all my serious work at home.

I am using a beige console with a Navarone Widget cartridge expander plugged in and three modules plugged in to the expander. To the right of the console is a Mechatronic 80 column card which is about 10 cm wide and as deep as the console and plugs into the I/O port. Plugged into the side of the 80 column card is a T-connector with a speech synthesizer plugged into its right. The total length of the plug-ins to the right of the console is about 30 cm. The console is modified with two 32 Kbyte static RAM chips replacing the two 128 byte static RAM chips on the 16 bit bus which provides not only the memory expansion but also gives physical memory from >8000 to >82FF. The modules plugged in at the moment are a Super Space cartridge, TI Extended BASIC and Microsoft Multiplan. Plugged into the back of the console is the power transformer and a cassette cable (not used very often). Into the back of the 80 column card is an 8 volt dc power supply (from a special regulator) and a cable to a monitor. The monitor is one of the Wang monitors with the tiltable base modified to remove the inverter in the synch line inside the monitor so that an external inverter is not needed between the 80 column card and the monitor. Out of the rear of the T-connector is the plug of the flat cable that goes to the PEBox. Sitting on top of the T-connector with its connections going into the T-connector, is a circuit to give a single load interrupt pulse each time the green button is pushed.

That is all on the desk top with the PEBox cable going over the back of the desk to the PEBox on a shelf below. This is a little inconvenient for insertion of disks but keeps the noise away from the user. Inside the PEBox I have two Horizon RAMdisks, one with 12 of 8 Kbyte static RAM chips and the other with 16 of 32 Kbyte static RAM chips. I have a Myarc hard and floppy disk controller card, a TI RS232 card and a TI P-code card. Connected to the disk controller card are two half height 5.25 inch floppy drives, one of them an 80 track drive (720 Kbytes) and the other a 40 track drive. There is also a hard disk of about 50 Mbyte capacity connected to the disk controller card and this sits on top of the PEBox. Connected to the RS232 via a Y-connector in the serial socket (DB25) are an Epson SQ-2000 ink-jet printer and a NetComm 123SA SmartModem. The PEBox is powered through an Autotransformer to lower the voltage to less than 200 volts and there is a large power supply for +5 and +12 volts for the hard disk drives which I made myself. This power supply is also used for the 8 volt supply for the 80 column card. This is my main system. Anyone else like to tell us all about their system?

PEBox card problems

One of our members, who shall remain nameless, has had some problems with the cards of his PEBox. Initially, he removed one of his RAMdisks (or was it inserted a RAMdisk?) with the power still turned on. This is almost guaranteed to cause major problems with the cards in the PEBox. This is because the power supplies to the cards are situated on the ends of the connector of each card. At one end (the pin 1 end towards the rear of the PEBox) is the +8 volt unregulated supply which goes to the 3 terminal regulator on each board to produce the +5 volt supply for the logic on the board. At the other end of the connector are the +18 volt and -18 volt unregulated supplies which are used on the boards to produce +12 volts, -12 volts and -5 volts where needed. The RS232 interface uses +12 and -12 volts while some of the ICs on the older cards like the 32 Kbyte memory expansion need -5 volts. If one card is mis-aligned in its socket it can cause adjacent pins in the socket to be connected together. This would be a disaster if the -18 volts was connected to any logic IC input or output which was only able to handle 0 to +5 volts. Unfortunately, the design did not have ground pins next to these unregulated supply chips in case this happened. That is what happened in this case. The pins adjacent to the -18 volt supply carry the MEMEN(L) signal and the CRUIN(H) signal. The MEMEN(L) signal almost always goes to a 74LS244 IC which is a buffer chip. This was the case in the 32 Kbyte memory expansion card, a TI disk controller card and the PEBox cable interface card with these chips being destroyed. This signal went to a 2732 EPROM in the CorComp RS232 card which was also destroyed and to a PAL chip in the CorComp disk controller card. Fortunately, both these chips can be replaced by re-programming new chips to replace them.

The other signal, CRUIN(H), goes to 74LS251 ICs in the TI disk controller and to a TMS9901 in the CorComp disk controller card but to TMS9902 and TMS9901 ICs in the CorComp RS232 card. It goes to the TMS9902 chips on the TI RS232 card. These are more expensive chips to replace and harder to get. The best thing to do is to be very careful with the cards in the PEBox. I have even known a PEBox to have been pushed back against an obstacle which caused the end of the RS232 card to hit something and move in its socket which caused the same problem to occur. It has also occurred with the two way interface which plugs directly into the I/O port with a socket with too much slack so that the card can be mis-aligned.

80 track floppy disks

I received the following communication from Pierre Garoche about the explanation I gave of his problem with trying to put 80 track and 40 track drives together.

Your explanations and advice in TND December 92 issue about quad density disk drives spur me on.
I decided to do some tests to have a better knowledge of the subject. As a result, I have understood that the MYARC FDC with an 80 track EPROM sends out one or two pulses to the heads stepper motor. This is dependent on two things: the information stored on sector 0 of the disk (number of tracks, density); and the setting of the switch in the MYARC FDC for the drive number used. Now I understand that I went wrong when I set my FDC to 40 track to connect to an 80 track drive. I am thankful for your information.

I made use of different programs to look at track positions. Their files are recorded on this floppy (with me, Geoff Trott).
Except for MARKSECT, the other programs may be used only with a MYARC FDC, because they link to the WD1770 directly. I do not know if you have such a controller.
I have used the following two step process. At first, store on the two sides of the disk rings set at regular intervals as are the lines on a graph paper. This is done by:

a- Initialize a floppy disk in DSQD mode.

b- Mark all sectors (except the first 4) with sector number in first 2 bytes and zero in other bytes. This is done using MARKSECT.

Next with the disk so set up, re-format it as DSDD and perform a new marking operation with a different pattern identification (AA and not 00). Now track positions may be found step by step with EXPLORER. They may be identified with TRR-S128. If you do not have a MYARC FDC it is possible to use DSKU to do a partial inspection. This will be done by changing byte >11 (17) from >28 into >50 in sector 0 of the disk, also change word >A (10) from >5A0 into >B40. Now with a sector editor, you may inspect the old even tracks on side 1 and the old odd tracks on side 2. You have to compute the sector numbers and DSKU option edit sector will display the sector. For the other sectors, an error is returned because on verification the track number does not match the sector number.

There are 2 special cases:

Starting from sector 0 you may go to sector >23 (35) without any error: From 0 to >11 (17) the sectors are track 0 sectors of the second formatting operation and from sector >12 (18) to >23 (35) they are track 1 sectors of the first formatting operation. The next disk sector is on track 1 of the second formatting operation and DSKU reads 1 as the track number which is not the correct value with heads at the 3rd step position.
Starting from sector >58E (1422) on track 79 side 1 you may go with no error to sector number >5B1 (1457) on track 80 the inner track of side 2.

Unfortunately I do not have a Myarc FDC so I cannot try out Pierre's programs. If someone wishes to have a copy of them, please contact me. I am glad that Pierre understood my explanation as it can be a difficult thing to write explanations clearly enough for others to comprehend them. At least it shows that I have at least one reader!

Myarc Hard disk controller bugs

I thought that I should say that my hunt of the bug that I wrote about last month has not got much further except that I have some very expensive equipment on the hunt, courtesy of my work. I am using a logic state analyser on the I/O port of the console to look at the addresses being accessed when disks are being read and written. I have discovered that I am using a version 12 EPROM which was supposed to solve a problem in the version 11 EPROM but introduced other problems. Other users are not using it apparently, which must be why no one else has the bug in their system. I have been using

this EPROM for a number of years with only this one problem that I am aware of. All I have to do is be careful about fractured files and everything else seems OK. I wonder if the other users of the Myarc hard and floppy controller card did not perservere long enough with version 12?

********** END OF ARTICLE **********

NOSTALGIA TIME

by Geoff Trott

This series of articles consists of my observations on the contents of the early TNDs starting with the earliest copies of our News Digest that I have in my possession. Assuming that you find that interesting, I am continuing with the series this month. Please stop me if you do not want me to continue. I will repeat my general disclaimer in case anyone reads this and gets the wrong idea. I am attempting to describe the look, layout and content of the newsletters without any critical intent. I will try to avoid using any adjectives which could cause offense and if anyone takes offense then that is purely their interpretation of the words and not my intention. I hope that makes it clear and that no one will be offended.

I have covered the issues from February to June 1983 in the previous two articles (except for the March issue) and now will look at the July and August magazines. These two, along with the June issue, form the first of the new format with a white cover, a front with the new logo, name of Sydney Newsdigest, table of contents and a theme picture and words. The back is divided into a top section with the information that currently appears on the mailing envelope, a centre section like the current left hand column of page 1 and a bottom section for notifying change of address and for asking for more information on various topics. The front cover remained like that for quite a while but the back cover only lasted three months before it changed. On the front cover of the July issue is a montage of photographs from the full day tutorial/workshop in June at St John's Church, Darlinghurst. Except for John Robinson and two other faces I cannot recognise, it is mainly backs of heads and fronts of TVs. There are more photographs inside the issue with a few more faces shown. I did not attend this day so I know that I am not there, but some of you may recognise the back of some of the heads. At this time Brian Lewis was Co-ordinator, John Robinson was Secretary, Terry Phillips was Treasurer, Peter Lynden was Education Co-ordinator, Manuel Constantinidis was Librarian, Shane Andersen was Editor and Graeme Hollis was at the end of the Crisis Line.

The July issue was 12 pages long (although the page numbering was not quite right with two pages numbered 9). In the Secretary's Notes, John wrote that there were over 50 members and guests at the full day workshop. The membership numbers had passed 350. Replies to the questionaire indicated that 46% were beginners, 43% had limited experience with 11% advanced programmers. Most members wanted to learn Extended BASIC but one third were interested in assembler. 79% thought that small group tutorials were a good idea. Terry Phillips agreed to also be acting Advertising Manager until the next elections.

Brian Lewis' Profile gives the dates and times for the monthly and regional group meetings and talks about the full day workshop. It sounds like good fun was had by all. Shane's editorial starts with an apology for those who did not get their newsdigest until after the meeting and so did not go to the workshop. Shane mentions two programs that have just come out, Demon Attack and Microsurgeon. He also gives some instructions for Flight Simulator, an Extended BASIC program.

Peter Lynden invited people interested in education to meet as a group. He then reviewed some of the available educational software. He thought that Beginning Grammar and Division One were very good programs whereas he did not like Reading Flight. David Liell wrote about getting a job with computers. He suggested that a good HSC result coupled with an interest and aptitude for computers is required. Getting familiar with the TI99/4A and learning to program should help. Younger Set is full of winners of various competitions. Steve Grassmyr won the workshop competition on Parsec with 1,068,600 points! Chris Reed scored 3343 points in Toad. Tunnels of Doom, Henhouse, Rabbit Trail and Medieval are mentioned as new games.

Russell Welham had an article on writing music using Extended BASIC. In this he explains some of the musical terms and gives the start of a program to play the tune "Mame" along with the words. There are also a few short programs to type in, including one in Spanish and one to use more than 16 colours.

The August issue had 16 pages with a front cover depicting satellite communication to Sydney. It also advertises a Compter Show at the Sydney Entertainment Centre. Inside, John Robinson wrote with a welcome to 16 new mwmbers and the information of a retail outlet in Newcastle which would give a discount to members. He mentions Softex Magazine from Doug Thomas in Melbourne, which I remember as a high class magazine which only lasted a year or so. The constitution was due to be printed in time for the AGM in November. Twenty copies of The Smart Programming Guide for Sprites from Miller's Graphics and 21 copies of the Tutor for assembler language were to be ordered. It is interesting how conservative the committee was in those days. There is also mention of a letter from Will McGovern (Tony's son) commenting on the fact that the data on the cassette tape from June was not readable. Will goes on to explain in some technical detail what the data should look like and what the TI99/4A does with it. Daniel Harris also wrote to John about providing a service to members to program their calculators for them for a price. I must ask him if he attracted any takers at that time. Someone wanted to sell a TI99/4 console for $100 and with Minimemory, Extended BASIC and various games modules $350 for the lot.

Shane wrote in International News about the receipt of the schematics for the Peripheral Expansion Box and a 220 page Software Directory. The Milton Bradley MBX voice recognition system system was announced, along with some sofware packages for it. Shane also had an article about Tronics Sales Corporation, which looks rather like a pyramid selling organisation trying to sell TI99/4As. I do not think it got off the ground in Australia.

Brian Lewis encouraged members to go to a regional group meeting. He also announced three software competitions which will be monthly if there are sufficiently good entries. Russell Welham continued his article on music and there was a short article on programming tips. There was a Techo Time article by Steve Williams on a Modification to General GC143 Receiver to Direct Video Input for TI Modulator Converted to Monitor. I am not certain exactly what it is about but it looks like feeding the composite video directly into the TV. The article does not really explain it very well.

On the centre pages there is an article on flowcharting which would help people learning to program except that it tries to be a bit humorous and may well end up confusing everyone. One page looks like a direct copy from the Rocky Mountains 99'er group about validating data on input in your program. The software in the issue includes a budget worksheet from the USA and Ground to Air Missiles from Col Christensen (Brisbane). That is about all for this month as the rest of the issue is filled with advertisements. That is a significant difference between the magazines then and now. Shane always had many pages of advertisements with lots of pictures and graphics. I wonder what a survey of our members would show today? I suspect that it would not be too much different from that in July 1983.

********** END OF ARTICLE **********

Regional Group Reports

Meeting Summary For APRIL

Banana Coast	11/04/93 Sawtell
Central Coast	10/04/93 Saratoga
Glebe	08/04/93 Glebe
Hunter Valley	11/04/93
Illawarra	20/04/93 Keiraville
Liverpool	09/04/93 Yagoona West
Northern Suburbs	22/04/93
Sutherland	16/04/93 Jannali

BANANA COAST Regional Group
(Coffs Harbour Environs)

We never miss meeting at Kerry Harrison's residence 15 Scarba St. Coffs Harbour, 2 pm second Sunday of the month. Visitors are most welcome. Contact Kerry 52 3736, Kevin 53 2649, Rex 51 2485 or John 54 1451.

CENTRAL COAST Regional Group

Regular meetings are normally held on the second Saturday of each month, 6.30pm at the home of John Goulton, 34 Mimosa Ave., Saratoga, (043) 69 3990. Contact Russell Welham (043)92 4000.

GLEBE Regional Group

Regular meetings are normally on the Thursday evening following the first Saturday of the month, at 8pm at 43 Boyce Street, Glebe. Contact Mike Slattery, (02) 692 8162.

HUNTER VALLEY Regional Group

The meetings are usually held on the second Saturday of each month at members homes starting at 3:15 pm. Check the location with Geoff Phillips on (049) 428 176. Note that after 9:00 pm this number is used for the ZZAP BBS which includes TI-99 information. Geoff.

ILLAWARRA Regional Group

Regular meetings are normally held on the second Monday of each month after the TIsHUG Sydney meeting, except January, at 7.30pm, at the home of Geoff & Heather Trott, 20 Robsons Road, Keiraville. A variety of activities accompany our meetings, including Word Processing, Spreadsheets and hardware repairs. Contact Lou Amadio on (042) 28 4906 for more information.

****
* LIVERPOOL Regional Group *
****

Regular meeting date is the Friday following the TIsHUG Sydney meeting at 7.30 pm. Contact Larry Saunders (02) 644-7377 (home) After 9.30 PM or at work (02) 708-1987 Liquorland Yagoona for more information.

APRIL (Good Friday)**********
My Place * Should be some *
34 Colechin St * new programs *
Yagoona West 2199 * from overseas *
****

NORTHERN SUBURBS Regional Group

Regular meetings are held on the fourth Thursday of the month. If you want any information please ring Dennis Norman on (02)452 3920, or Dick Warburton on (02) 918 8132. Come and join in our fun.
Dick Warburton.

SUTHERLAND Regional Group

The second meeting for this year, in February, was again an interesting meeting with a number of topics being covered. Not least of all was our involvement in the monthly paste up of the club magazine.

Some time was spent in experimenting with the various pitch styles available through the "Stylealine" programme, to achieve the 55 column format for the magazine.

We put a couple of new disks through their paces including T.I. Artist Fonts and some assorted games disks which included Solitare/Poker and Snakes and Ladders. Both were recently released through the club shop.

Another disk included TIPS pictures converted to Page Pro format, by Alf Ruggeri, which included a Dimensioner. The dimensioner documentation was found to be incomplete. Joe D'ambra also wrestled with a corrupt Banners disk, but to no avail.

Regular meetings are held on the third Friday of each month at the home of Peter Young, 51 Jannali Avenue, Jannali at 7.30pm. Peter Young

---

TIsHUG in Sydney

Monthly meetings start promptly at 2pm (except for full day tutorials) on the first Saturday of the month that is not part of a long weekend. They are held at the RYDE INFANTS SCHOOL, Tucker Street (Post Office end), Ryde. Regular items include news from the directors, the publications library, the shop, and demonstrations of monthly software.

APRIL MEETING - 3rd APRIL

As stated elsewhere, Ross Mudie is bringing in his updated version of the brilliantly programmed and designed train set-up that is totally run by the computer. The shop will be opened, as usual. See Percy's article about available items. Any new software from overseas will be demonstrated. If you have a piece of hardware that needs Geoff's attention give him a call before the meeting so he can bring the necessary gear in to examine and repair it. We all enjoy the fellowship and activites each month. Come along and join us.

****

The cut-off dates for submitting articles to the Editor for the TND via the BBS or otherwise are:

May	11th April
June	9th May

These dates are all Sundays and there is no guarantee that they will make the magazine unless they are uploaded by 6:00pm, at the latest. Longer articles should be to hand well before the above dates to ensure there is time to edit them.

****

BEGINNERS EXTENDED BASIC WEEKEND
by Ross Mudie, 6th March 1993

At the November 1992 meeting, a number of newer members who were having problems getting started with their TI99 computers, expressed the view that more help was needed for the beginners in the club. An increased number of beginners articles was started in the next TISHUG News Digest and a two day beginners tutorial weekend on Extended BASIC was proposed.

The date for the tutorial weekend is 17th & 18th April. At the deadline date for advice of who wanted to attend, (the meeting of 6th March), insufficient people had indicated interest for the weekend tutorial to go ahead.

The deadline for the tutorial weekend has now been extended to the April 3rd meeting. Beginners please, if you are REALLY SERIOUS about getting started with your computer then you should (1) ring Dick Warburton at home on 02 918 8132 to register your interest and (2) come to the TISHUG meeting at Ryde Infants School on 3rd April to communicate what you want to get out of the tutorial weekend.

Is the increase of beginners articles and the organisation of the tutorial weekend what the members want?

Please provide feedback to the authors of the articles if they are what you want, especially if you want more of the articles & enrol for the Extended BASIC tutorial weekend on or before 3rd April.