Ottawa 065
8806

The Ottawa T.I.99/4A Users' Group

VOLUME 7 NUMBER 06.........JUNE 1988

DON'T FORGET THE MEETING -- June 7, 1988

P.O. BOX 2144, STATION D, OTTAWA
*** ONTARIO, CANADA K1P 5W3 ***

COMING EVENTS

June Meeting:	June 7, 1988	Merivale High School Raffle! Format flip! Come on out!
Assembly Workshop:	June 15, 1988 7:30 p.m.	Dick Piché's home Contact Bill Sponchia for further information.
Summer Barbecue:	June 25, 1988	Marg & John O'Connor's home. Bring all your own everything. Map page 5; further details in President's Two Cents Worth.
Newsletter Deadline:	August 14, 1988	(or any time before that!)

EDITOR'S NOTES
from
Ruth O'Neill

The cover art comes to us from Lucie, again, as does the map to the O'Connor's home on page 5 of this newsletter. There will be no newsletter for July and August, as usual, so the next newsletter deadline is August 14. I'd be happy to hear from anyone sooner, of course. I'll be away the last week of July and the first week of August, so you'll have to contact me either before or after that. Until September, then!

DM1000 Warning:

Most importantly of all this month, we would like everyone to know that the Ottawa TI-99/4A Users' group recommends that anyone using DM1000 3.6 or higher go back to 3.5. It is the most stable version at the moment, and later versions may contain potentially serious bugs. We apologize to anyone who may have experienced them, and hope that they will enjoy using 3.5 until our next release. Version 4.0, for example, is an escaped beta copy and should not be used under ANY circumstances. We hope to have a new version (thoroughly beta-tested) ready to go in the fall.

Still on the subject of DM1000, if you have received or know of anyone who has received a copy of DM1000 from Texcomp with their Explorer package, we would like you to know that that version is very old and outdated. We would like to ask your help in passing the word that they can get a much better version directly from us by writing to the club address on the cover of this newsletter, and sending a diskette, mailer, and return postage. A contribution is always appreciated, but not necessary; try it out before you decide.

While as a club we tend to focus on DM1000 as our fairware, we should also remember that we have other fairware -- for example, JP Graphics, Steve McWatty's Graphic labelmaker, Maro Tomietto's Sideways for Multiplan, Some assembly language utilities from Art Green, and still more. Ask about them! You'll probably be surprised. And the next time someone asks about fairware by the club, tell them about more than just DM1000, because while we are proud of it, we have much more to offer.

Recently, I've been reading some very harsh words against archiving; it seems that many people feel that there is no place for archived files other than on BBS's where downloading a compressed file will save the user time and/or money. Some don't even feel it should be used except where connect time is being charged. I think that's a little extreme, myself. On a busy board such as ours, compressed files make people's visits a little shorter, so that it's easier for others to get on. Archiving does have legitimate uses, though, other than on BBS's. More programs today are becoming larger, and have more files attached to them. Have any of you ever experienced the frustration of going to use an eagerly-anticipated program, and finding one of the pieces missing? Chances are, if whoever had given or sent it to you as an archive, you wouldn't be missing those pieces. So that's another use -- keeping all the pieces together. Still a third is saving money on distribution of diskettes. This is, apparently, one of the most loathed uses for archiving, but those who view it that way are being somewhat selfish in my view. "An extra diskette is only 25 cents, so why archive? Why not send another diskette?" they say. Allowing that, (although here in Ottawa the cost is closer to 60 cents) there is also, perhaps, an increased cost in postage, the extra time involved in making more copies, the extra labels to make and affix... and anything else involved in preparing one disk, doubled. This wouldn't be significant for one or two diskettes, which is all that the receiver sees. What if, though, the sender is not sending to ONE receiver, but hundreds? or a thousand? It quickly adds up to hundreds of dollars, which is considerably more than just 25 cents. While I don't think everyone should archive everything they have, I don't think archiving is quite the monster that some people seem to think. It is a tremendous help to all of us in this community, and we should be happy to see people using it.

As an experiment last month, the articles from the May 1988 newsletter were uploaded to Compuserve, in both archived and ascii formats. While those readers don't see the artwork or paid advertising, we thought that the distribution of information is the most important aspect of this newsletter.

Since we simply can't afford to exchange newsletters with every users' group on the continent (while we would like to, of course), we are offering this as an alternative way for other users' groups to read our articles. We hope that the newsletter will be downloaded and distributed by other BBS's as well. It seemed to be very well received last month, so for the time being we will continue to upload the text from this newsletter. We are, of course, still continuing our newsletter exchange. This is intended to expand our readership, not change our distribution methods.

Last of all, but certainly not least, I'd like to say goodbye to Bob Boone. Well, actually, it isn't something I'd like to do, but must. For those who don't already know, Bob has been transferred to North Bay, and will be leaving Ottawa in July. I'm sure North Bay is very happy to hear that, but Ottawa is not. It will seem strange not to see his store set up at every meeting, and we'll suffer some from the lost access to his ideas, but most of all, I think, we'll simply miss him. I hope that he'll be able to visit us often, but I'm sure it won't be as often as we'd like. Goodbye, Bob, and come back soon.

The President's Two Cents' Worth
by Michael Taylor

At our June meeting, in view of the enthusiastic response to a proposal by Archie Simmonds, we will hold a raffle. The prize will be a gift certificate from Computer Download Unlimited. The value of the certificate will be equal to the sum collected from ticket sales during the meeting. Tickets are 50 cents each, or 3 for a dollar. The more tickets sold, the greater the value of the prize; so come on out to this, the last meeting before our summer recess, and buy a chance or three towards that piece of hardware or software you have coveted.

Another proposal that elicited favorable response was one to reverse the program format at the general meeting. This will be implemented, on a trial basis, at the June meeting. The change will see the demonstrations presented at the beginning of the evening and the business deferred to the end. This will allow us to clear out of the school by the official deadline and so endear us with the janitorial staff. As well, this arrangement will preclude the necessity for a mid-session break. The demonstrations will begin `on time', at 7:30pm. All-in-all, we anticipate accomplishing more within our allotted time.

As for the demonstrations, we will be calling upon the authors in the Pro category of the software contest to show their stuff.

The club has received some bare printed circuit boards for constructing the Horizon RAMdisk card. These are being offered to members on a first come, first served basis. The price is $45 each. Please contact me if you would like one.

We are planning a series of workshops dealing with the construction of the RAMdisk card. Dates are not set yet.

In regard to the Pool Party & Barbecue at the O'Connor's on June 25th, please remember: no glass within the pool area! Plastic tumblers will be available for those who forget to bring their own. Arrival at the O'Connor's can be any time after 1:00pm. The gas grills will be fired up at around 4:00pm. Since this bash will be our last formal meeting until September, come on out and enjoy yourself -- and bring the family!

BROWSING THE LIBRARY
--with DAVE MORRISON

As the June meeting will be the last meeting prior to the Summer break, I intend to enlarge my (apparently well-received) multiple disk offering. In addition to the dozen or so disks that I offered at the May meeting, I shall be providing a selection of Basic (TEII) and XB speech programmmes, some graphics and art programmes, plus whatever else that I consider to be of interest.

Documentation for perusal at the meeting will again be provided and each disk will be supplied with a catalogue of the contents.

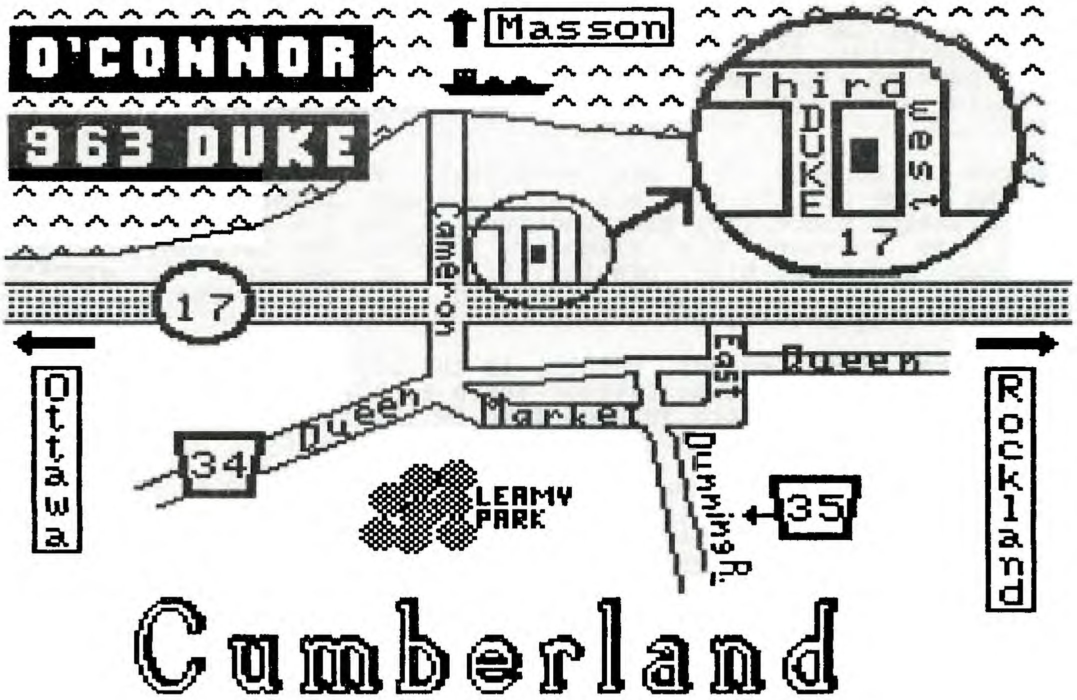
I wish to emphasize that the $3.00 fee paid to the Library is a copying charge only and does not absolve the purchaser of the responsibility, in the case of "Fair/Fare-ware", to compensate the author/programmer for his/her endeavours. Only you can decide whether the programme is worth a donation. Please review each programme carefully and objectively and act accordingly. In other words -- PAY WHAT IT IS WORTH! I am sure that, at the least, a letter of appreciation and encouragement would gladden the heart (if not the pocket-book) of the programmer!

The Library will remain open during the Summer months and I'm only a telephone call away!

Have a very pleasant and relaxing Summer. Dave

How to get to the pool party!

The map below should help you find your way to the O'Connor's home for our summer barbecue. From Ottawa, head east on Highway 17, turn left at Cameron (look for the flashing yellow light), take the first right (Third), then the first right again, and the house on the left should be 17 Duke, where Marg and John live. There are only 2 houses there, one on each side of the road, so there's no excuse for you not to be there! See you!



Legends 1.1 -- A review by Ruth O'Neill
or
There and (maybe) back again

Are you a Tunnels of Doom fan? If so, you'll LOVE Legends. Actually, even if you don't enjoy TOD, you'll probably like Legends, because although it is a similar type of game, it's in another class. It's much more challenging, interesting and sophisticated, and will require more thought and planning.

Legends is set on fairly large island, with different types of terrain: Grasslands (mildly dangerous), forests (dangerous), swamps (VERY dangerous), mountains (impassable), and water. Scattered over the Island are inns (good for resting players), icons with various information, teleporters, and ruins. The ruins are maze-like dungeons, either inhabited by monsters or riddled with traps. Other monsters roam the land above them. Your quest team consists of four players: A fighter, a ranger, a wizard, and a cleric. The game comes with a team assembled and ready to go, but I prefer to build my own characters and name them myself, an easy procedure in Legends. While the process is easy, it shouldn't be taken lightly: your choice of players will affect many things throughout the game, not the least of which is how easily you'll survive conflicts with monsters.

The quest in Legends 1.1 is to close the Western Portal through which the Ashtar Creel has loosed all the dreadful monsters you'll have to deal with along the way. To do this, you'll need certain items which are hidden somewhere on the island or in the dungeons. Before you go too far from the city of Wizard's Rock, however, you'll need to train your players and teach them some spells, or they won't live long. To train your players at the adventurer's guild, you'll need to accumulate experience, and gold to pay for the training. Experience is gained by fighting monsters (and winning). Gold is won in the same fights, or you may find it in the dungeons. The spells you gain by training your players are extremely important in battle. With them you can inflict damage on your opponents, heal your players, protect yourself, and a number of other useful things.

One of the best improvements Legends 1.1 made over 1.0 is that players can strike more than once each turn. This makes battles go by faster, but with about the same results. Of course, the monsters can also swing twice each per turn. Fair's fair, after all. After I had trained my players some, I noticed that I could then hit THREE times each. I felt pretty cocky, until I wandered into a forest and met the monsters that could also hit three times each. Balancing this is the spell "SLOW". Once you have trained enough, you can cast a "slow" spell at the monsters you are fighting, so that they can only hit once each. Unfortunately, they can also cast "slow" at you. The only way to reverse this is by casting a "dispel magic" spell, which, unfortunately, your wizard doesn't learn until level two. Just staying alive at level one is a bit of a challenge!

The graphics in Legends 1.1 are excellent. The island is interesting to explore, and some of the monsters are fabulous. I think my favorites are the spectres, although since they are dangerous, I try not to stay around them too long. All the monsters are slightly animated for battles, too. They swing clubs, swords, and paws, bite, breathe fire, or whatever is appropriate to them. It's easy to tell by watching them and listening to the sound effects how badly you are being beaten.

What I like most about Legends, though, is the sly sense of humour evident in both the game and the documentation: Where else would a turnip be a dangerous monster? It is, though. Death Turnips live in the forest, and although you may be tempted to laugh, you'd best be careful.

I'm sorry I don't have more room here to describe it fully -- it is a difficult game to describe, and it's also hard to explain how whole weekends can disappear while I'm playing it. It is a lot of fun -- intriguing and exciting at the same time. I'm sure you'll enjoy it. Do yourself a favour and ask Bob Boone (Computer Download Unlimited) if he has any in stock, or you can order directly from Asgard Software, by sending $27.95 (US) plus $0.75 Postage (orders over $50.00 need send nothing for postage) to: Asgard Software, P.O. Box 10306, Rockville, MD 20850. (phone 301-559-2429)

One Warning: Before going off on your final quest to battle the Ashtar Creel, check line 1200 in the LGDN/MON program. If it includes the statement "OK=K", change that to "OK=F", substituting an F for the second K. This will allow you to finish the game properly. Enjoy!

(In other news from Asgard, Calendar Maker is now available in any of four versions: English, French, German, or Spanish. If you bought an English copy, and would prefer another language, you can send it to Asgard, along with to return postage, with your request for whichever language you would prefer.)

HINTS, TIPS & ANSWERS
by Bill Sponchia

The other day, I got a phone call which made me go back through some old copies of our Newsletter. It got me to wondering how many of you out there actually save this or any other group newsletter. If you aren't, then you are missing out on one great source of information. It was amazing how much was in the few that I looked at (from 1985 and 1986) which now appeared interesting to me... at the time I had no interest, so just sort of breezed over it. In the coming months I will attempt to rehash some of the tid-bits.

Although I save our complete newsletter I find that saving all the group newsletters becomes too much, so what I do is when I read them I mark sections that I want to copy. Later I take the copies and then file these. It not only is a lot less bulky, but also makes it easier to find the information when I want to review it.

Here are some interesting things that I have read as I have gone through the many newsletters now in our possession. Some are old hat for many of us but it's amazing that general knowledge for some is the news of the century for others.

Multiplan

Hint - To save time when inputting information (data or formulae) turn off the "recalculation". Press "O" for Options and then "N" for No. After all entries are made, the entire sheet can be recalculated by pressing FCTN 8.

Hint - To recalculate a single cell when the RECAL is off, set the pointer to the cell you wish to recalculate, press "E" for Edit then press ENTER. Only that cell will be recalculated. Please note that this might not give you a correct answer if it depends on information in other cells because the other cells will have not been recalculated and thus have incorrect data in them.

Extended Basic

Hint - If you are tired of waiting for that long XB program to start operating after you RUN then possibly you require the use of "prescan". It's a feature that TI hid away in the Addendum to the TI Extended Basic Manual (page 7). Don't use it until the program is bug free.

Hint - Ever accidentally press FCTN = (Quit) and loose all the typing you have done? Well, you can disable the Quit Key by entering the following:

```
CALL INIT::CALL LOAD(-31806,16)
```

To re-enable it:

```
CALL INIT::CALL LOAD(-31806,0)
```

TI-Writer

Hint - When using the formatter commands, remember the following:
the ".CE" (centering) command will centre your text between the ".LM" (left margin) and ".RM" (right margin), not the middle of your paper.

That's all for now folks....this is Bill signing off....No Carrier!!

PROGRAMMING HINTS
By David Caron

It seems I have run out of comprehensive things to talk about (stuff I know, but you may not). Therefore, I will devote this article to interesting things I have found out in programming.

The following hints refer to the Extended Basic programing environment.

1. Have you ever noticed that when you are in the immediate mode and have created a sprite, that it only lasts for about half a second? Well, now you can keep that sprite there forever while you are in immediate mode. You can edit, list, change and delete program lines without affecting the lifespan of that sprite. In fact, this is also true for character colour changes and character definitions, as well as screen colour changes.

Let's say you have gotten tired of looking at your extended basic program in black letters on a white screen, and wish to change the foreground colour to white and the background colour to dark blue. To do this, you need only execute the commands:

```
FOR CC=1 TO 14 ::CALL COLOR(CC,16,1)::NEXT CC::CALL SCREEN(5)
```

Unfortunately, this effect will only last for a very short period of time, BUT if you add the statement:

```
ACCEPT AT(24,1):A$
```

to the end of the first line (with the :: naturally), and then press BREAK after typing run, you will be returned to immediate mode without Extended basic resetting the colours.
This strange phenomenon will continue as long as you do not try to execute any commands other than for LIST or RUN. You can still edit or change lines. Unfortunately, this effect does not work while ACCEPT at is being executed in running mode.

2. If you have ever wanted to produce an everlasting sound (one longer than 4.150 seconds, it is possible using the ON ERROR routines. This is, however rather complicated.
Let's say we wish to have to following call sound play endlessly:

```
CALL SOUND(-4150,117,1,116,1,-3,1)
```

The following program will do this.

```
10 ON ERROR 100
20 CALL SOUND(-4150,117,1,116,1,-3,1)
30 CALL SOUND(-10000,110,1)
40 END     !(or to the rest of your program)
100 RETURN NEXT
```

When line 30 is executed an error occurs, since -10000 is an invalid duration. Line 100 is executed because of the error. RETURN NEXT then causes execution to start at the statements after the error. By this time, the computer has "forgotten" to turn off the sound generator after 4.150 seconds and therefore the sound continues on endlessly. It can be stopped simply by executing another call sound, or pressing Function 4, causing a break in a program which will produce a low tone, which overrides the previous sound.

3. This last hint is not something I discovered, but something Art Green told me would save me memory, in that it would reduce the size of a program. The idea is to replace all of the constants used in a Basic program with variables. Therefore if you had a statement:

```
10 CALL HCHAR(1,1,32,736)
```

memory would be saved by using this technique.

```
1 @1=1
2 @32=32
3 @736=736
10 CALL HCHAR(1,1,32,736)
```

Although it takes memory to represent the line @1=1 , if many "1"'s were replaced with @1, then you would end up benefiting from extra free memory.

These are all the interesting things I have come across that I have not seen mentioned in previous newsletters. Lucie Dorais is doing a great job of explaining the rest of the features of EXTENDED BASIC.

FAST
EXTENDED BASIC

LUCIE DORAIS

As promised in the last issue, here is the SORT module of my STAMP DATA BASE program: just add the following lines to the program published last month.

But, first, there are a few things to change in the main program. At the end of line 120, add the following statements:

```
120 ... :: CALL CHARPAT(120,A$) :: CALL CHAR(37,A$)
```

This will redesign the "%" as a "y"; we need it if we want to write some text in lower case with the "y", because in line 140 we redesign the normal "y" (ASCII 120) as a graphic character used in the display of the stamp. We will also use some new variables in the sort function, so please add those three variable names in the pre-scan line 150: TN, Y, Z. (Note: in the May issue, the pre-scan statements came out with two "at signs", of course this is a boo-boo from a modified Formatter, there should be only one, lines 150 and 930.)

```
390 TN=N :: N=0 :: S$(0)="SRT" :: GOSUB 900 ! get "sort" record
400 DISPLAY AT(20,1):" Enter ""***s"" in the field    %ou want to
    SORT on.":"":" To return to Menu,  enter     nothing
    ever%where."
410 CALL A(4,13,15,COU$):: CALL A(6,13,7,DA$) ::
    CALL A(13,14,14,COL$)! get field to sort on
420 S$(0)="0  |"\COU$\DA$\RPT$("    |",3)\COL$\" |   |"
430 P=POS(S$(0),"*",1):: IF P=0 THEN N=TN :: GOSUB 900 :: GOTO 260
     ELSE GOSUB 870 :: DISPLAY AT(22,9):"SURTING..."
440 S=POS(S$(0),"|",P):: S=S-P :: D$=RPT$("~",65):: K=1
450 K=2*K :: IF K<=T THEN 450
460 K=INT(K/2):: IF K=0 THEN 510
470 FOR X=1 TO T-K :: Y=X :: IF POS(S$(Y),"~",5)<>0 THEN S$(Y)=D$
480 Z=Y+K :: IF SEG$(S$(Y),P,S)<=SEG$(S$(Z),P,S)THEN 500
490 A$=S$(Y):: S$(Y)=S$(Z):: S$(Z)=A$ :: Y=Y-K :: IF Y>0 THEN 480
500 NEXT X :: GOTO 460
510 FOR X=1 TO T :: IF POS(S$(X),"~",5)<>0 THEN T=X-1 :: GOSUB 920
    :: GOTO 530 ! stop at first deleted rec. (i.e. pack)
520 S$(X)=SEG$(STR$(X)\"  ",1,3)\SEG$(S$(X),4,65):: NEXT X
    ! re-number each record
530 N=1 :: GOSUB 900 :: GOTO 260 ! back to menu, show rec. #1
```

To get to the Sort function, you must press the "*" from the Menu; why, do you ask, did I choose that character? Well, the [S] was already taken by the save function; also, since sorting is a time-consuming operation, better use a key that you would not press by accident!

The first thing we do, in line 390, is to store the value of N (record on screen) into TN, temporary N: we will need it if we make a mistake and want to get out of the sorting input screen. Line 400 displays the instructions for the sort function; don't forget to type a "%" instead of a "y".

Line 410 uses the CALL A sub to tell Tex which field to sort on; you can sort only one field at a time. We have to use the simpler CALL A because it is the only one that accepts the asterisk as a character... Since all our fields are not useful to sort on, we use only three CALL As: for Country name, Date, or Color; we just repeat the parameters from lines 230-240 of the main program. Just enter "*"s (only one will do as well) in the field to sort on, nothing in the other two; if you put an asterisk in more than one field, the program will sort on the first one only.

Line 420 uses record #0, i.e. S$(0), to build the "sort string" (if you outsmarted me when typing the program in May by adding OPTION BASE 1 to it, take it out now!); this string is identical to the data ones, so it will be easy for Tex to determine which part of each string to sort. Note that we replace the unused fields by spaces; since three consecutive fields have four characters in size, we use the RPT$ statement. Compare the sort string (Color is the field to sort on) with a real data one:

```
Rec|     Country      | date   |Sc# |Pic |Face|      Color      |S|Val.
----------------------------------------------------------------------
0   |                 |        |    |    |    |***              | |     |
15  |CANADA           |1952    |320 |A136|7   |BLUE             |N|40   |
```

We then get the position of the first (or only) "*" in the sorting string; if none, then back to menu, otherwise GO SORT! In line 440, we look for the first delimiter "|" after the asterisk, so that Tex does not kill itself sorting more characters than it should. We then calculate S, size of the string to sort. D$ is a full line of tildes; Tex will replace each deleted record with it; when we sort on any field, the deleted records, being all "~"s (with an ASCII number above any alphabetical character) will all be placed at the end of the list, and the "packing", i.e. the actual deleting, will be much easier to do. K=1 is the beginning of the sort function proper. Note that I re-use the K variable (usually used for KEY), and I also re-used the S (normally STATUS), since both are not needed in that module of the program: this saves memory, but I admit it can be confusing.

The sorting function used here, which ends at line 500, is called the SHELL SORT, after the man who designed the algorithm. I took it from Regena's "Guide to the TI-99/4A", page 295; she explains how it works, but I admit I do not understand it all, so just have faith, it works! It is a binary sort, that breaks the main list into smaller ones, easier to deal with. It is not instantaneous, since 50 records take about one minute to sort, but much faster that the BUBBLE one.

Now that all our records are sorted, we have two problems: the deleted ones are still there, although now placed at the end of the list; and our record numbers, part of the data strings, are all mixed up. So, lines 510-520 do two things: if the next data string, S$(X), is a deleted one (it contains tildes), get out of the re-numbering function, but do not forget to bring the total T down to the last good data (X-1). If, on the other hand, it is still valid, renumber it sequentially by replacing the first three characters by the correct record number. When all is done, go back to menu and show record number N=1.

Since I still have a lot of room to write, I added another feature to the May program: when you want to [G]et a record, it is sometimes easier to find data than to remember the correct record number, so I added the following lines in the Get function. Before you rush into typing them, please note that the former line 350 has now been renumbered 356, without any change. Use the EDIT function and the REDO key to make the modification.

```
350 GOSUB 870 :: DISPLAY AT(20,2):"Press:  D": :" [R] to get a
    record number":" [D] to get data"
351 CALL AS(20,10,1,A$,"RD"):: IF ASC(A$)=82 THEN 356
352 GOSUB 870 :: DISPLAY AT(21,1):"DATA TO GET: " ::
    ACCEPT AT(21,14):D$ :: Z=1
353 FOR X=Z TO T :: IF POS(S$(X),D$,5)=0 THEN 355 ELSE N=X ::
    GOSUB 900 :: DISPLAY AT(23,1)BEEP:"THIS RECORD? Y"
354 CALL AS(23,14,1,A$,"YN") :: IF ASC(A$)=89 THEN 260 ELSE
    DISPLAY AT(23,1):"" :: Z=X+1 :: GOTO 353
355 NEXT X :: DISPLAY AT(23,9)BEEP:"NOT FOUND !" ::
    FOR X=1 TO 300 :: NEXT X :: GOTO 260
356 A$="GET" :: GOSUB 890 :: IF P THEN N=P :: GOSUB 900 ::
    GOTO 260 ELSE 260 ! get record by number
```

Sorry, there was no room to add another item to the menu, so Tex will always ask you to choose between Record number or Data, the latter being the default. Again, we need to check the Ascii value because of the automatic addition of the delimiter in the CALL AS sub. If we press "R", we go to line 356, ex-350.

Line 352 asks for the data to get; since the CALL A sub adds a "|" at the end of the string, we would have had to strip it, a tedious programming job, so we use a straightforward ACCEPT statement. Starting with record Z=1, Tex will look for our string; when it finds something, it displays that record by giving the value of the X counter to N before exiting the loop. It then asks you if it is the record you want; if not, the value of Z is incremented by one and given to the X loop counter to continue the search. If it reached the end, that is, the NEXT X has run its course, it displays a brief message, then goes back to the menu; note that variable X is re-used for the delay counter.

About line 356, which used to be 350: the statement "IF P" is just a quick way to say "IF P<>0", very useful in a case like here where there are only two courses to take, whether P is equal to "0" (return to menu) or not (get the record number P).

Do I still have room? How about a module to find out the total value of your collection? Truly, this is NOT the best way to write a program: everything should be planned in advance. Anyway, this is the last column for this year, so please forgive me for all those afterthoughts... And, as a summer project, how about reworking the program so that "GET DATA" and "COUNT TOTAL VALUE" can be called directly from the main menu?

Again, don't rush: you will have to modify line 850.

```
850 GOSUB 870 :: DISPLAY AT(21,8):"QUIT (Y/N)?  N": :"[Enter ""T""
    to get total cat.value before quitting pgm.]"
851 CALL AS(21,21,1,A$,"YNT"):: IF ASC(A$)=78 THEN 260 ELSE IF
    ASC(A$)=89 THEN END
852 GOSUB 870 :: Z=0 :: DISPLAY AT(21,3):"TOTAL VALUE: $"
853 FOR X=1 TO T :: A$=SEG$(S$(X),61,4) :: Y=VAL(A$) ::
    IF POS(A$,".",1)<>0 THEN Y=Y*100
854 Z=Z+Y :: DISPLAY AT(21,17):Z/100 :: NEXT X :: DISPLAY AT(24,3):
    "Press an% ke% to quit..."
855 CALL KEY(0,K,S):: IF S=0 THEN 855 ELSE STOP
```

You first start by splitting line 850 into 850 and 851 (again, use the REDO function); and you will need to change the line of the DISPLAY and ACCEPT statements from 22 to 21, to leave room for the extra text (watch for those "%"s). In line 851, add "T" to the validating string, and modify the IF statements as shown.

To count the total value Z, we get that portion of each entry that corresponds to the catalogue value; that is found starting at character 61 of the string, and has a maximum of four characters. We then use the VAL function of Tex to get the numeric value (Y) of the string. Since your values can be entered in more than one way (for example, 50 cents can be 50 or .50, and one dollar can be 1.00 or 100), the next statement looks for a period (just be consistent when you enter your data). If there is one, multiply the value by 100.

You will actually see the total growing as each value is added to the total; of course, we have to divide it by ten before showing it. When Tex reaches the last record (T), you can contemplate your fortune until you press a key.
Well, this is the end for this year; I need a rest! Enjoy your summer -- I know I will enjoy mine. I already have some plans for next year, "top secret until all details are finalized"...

Don't forget the trading post!

It's here for you!

TI BASIC continued from May
by **Steven Shaw**

JOYSTICK & KEYBOARD CONTROL PROGRAM

```
100 CALL CLEAR
110 RW=12
120 CL=16
130 CALL HCHAR(RW,CL,42)
140 CALL KEY(1,K,ST)
150 CALL JOYST(1,CR,RR)
160 CALL JOYST(2,CR2,RR2)
170 IF (CR+2*RR<>0)+(CR2+2*RR2<>0)+(ST<>0)THEN 180 ELSE 140
180 IF ST THEN 240
190 CALL HCHAR(RW,CL,32)
200 RW=RW-RR/4-RR2/4+(RW=1)*((RR=4)+(RR2=4))-(RW=24)*((RR<0)+
    (RR2<0))
210 CL=CL+CR/4+CR2/4+(CL=1)*((CR<0)+(CR2<2))-(CL=32)*((CR>0)+
    (CR2>0))
220 CALL HCHAR(RW,CL,42)
230 GOTO 140
240 CALL HCHAR(RW,CL,32)
250 RW=RW-(RW<>1)*((K=5)+(K=4)+(K=6))+(RW<>24)*((K=15)+(K+1=1)+
    (K=14))
260 CL=CL-(CL<>1)*((K=4)+(K=2)+(K=15))+(CL<>32)*((K=6)+(K=3)+
    (K=14))
270 CALL HCHAR(RW,CL,42)
280 GOTO 140
```

ACCEPT AT ROUTINE IN TI BASIC

REMINDER: Set R & VR to desired row-column The input is placed in variable IN$ Use GOSUB to enter this routine when it is required Remove line 160 if you do not wish to be limited to entry of numbers 0 to 9.

```
100 IN$=""
110 CALL KEY(0,K,S)
120 CALL HCHAR(R,VR,30)
130 CALL HCHAR (R,VR,32)
140 IF S<1 THEN 110
150 IF K=13 THEN 250
160 IF (K<48)+(K>57) THEN 110
170 CALL HCHAR(R,VR,K)
180 VR=VR+1
190 IF VR<33 THEN 230
200 VR=32
210 K=13
220 GOTO 250
230 IN$=IN$&CHR$(K)
240 GOTO 110
250 IF IN$="" THEN 110
260 RETURN
```

The next program is an ACCEPT AT routine in TI BASIC. A PRINT AT routine can be found in VINCE APPS book of TI99/4A programs.

The INPUT command causes the screen to scroll, and in the middle of a game with a complex screen display that can be disruptive!

The required routine will allow you to fill a variable string and place the input onto any desired part of the screen.

The initial screen location is held in the variables R(row) and VR(column).

The required input is to be placed in a string variable IN$. To ensure the variable is 'empty' it is cleared at the start of the routine.

The string is filled by means of a series of CALL KEYs, terminated with the ENTER key (which gives a code of 13).

For user confidence the cursor (character 30) is flashed at the position the input will appear at. This comes immediately after the call key. If required a CALL SOUND could be inserted just before the CALL KEY to provide the usual 'input' tone.

If no key is pressed, the cursor will just flash.

When a key is pressed, a check is made to see if it is the ENTER key, to terminate input (Key code 13). If ENTER has not been pressed, you can check to ensure the key falls within a required range.

In the sample given, only the number keys 0 to 9 are accepted as inputs (ASCI codes 48 to 57). If the choice of keys is not so neatly in sequence, you may use:

```
   IF POS("ESDX",CHR$(S),1)<1 THEN 110
```

If the key pressed is not E, S, D or X in this example, the key is not accepted and the program returns to the CALL KEY.

If the key IS accepted, the letter is placed on the screen in the appropriate location, and the value of the column is increased by one.

A check is then made to see if the input has gone past the end of the screen: if no check was made, a BAD VALUE would be possible.

In this example, if the column exceeds a value of 33, it is reset to 32 and an automatic ENTER is inserted to terminate the input. You may prefer to substitute GOTO 110 instead of ending the input : this alternative places the cursor back on the last position (eg at screen right).

If the key is accepted, the character the key represents is added to the input string and the program returns to the CALL KEY.

Once ENTER has been pressed, a check is made to see if an input HAS been made (a nul input string could cause your program to crash). If no entry has been made the program returns to the CALL KEY.

If all is well, you are allowed to RETURN to the place in your program you left with a GOSUB. The input now lies in variable string IN$ for you to manipulate as you wish.

As the example program has limited the input to digits, a numeric variable can be set by using N=VAL(IN$) in your program.

Additional programs may be found elsewhere in this book, Try to see how they work - can you improve them!

A quick routine for right justification (useful when using columns of numbers)-the variable to be printed is in the string variable A$ (use STR$(4) etc to convert numbers to strings) and the right column in variable RC:

```
   X=RC-LEN(A$)
   FOR C=1 TO X
   A$=" "&A$      (this adds one space in front of A$)
   NEXT C
   PRINT A$
```

SIMPLE TIPS:
A=B^2 takes longer to process than A=B*B.
A=20000 uses more program memory than A=2E4

DATA COMPRESSION

When memory is limited, any technique which allows you to fit a quart into a pint pot is useful.

A number of data compression techniques have been evolved, some of them quite complex, but we shall deal with the subject only briefly here.

First remember that the TI99/4A stores:
NUMBERS in the number of digits in the number plus 2 bytes.
STRINGS in the number of characters in the string plus two.
NUMERIC VARIABLES in the number of characters in the variable name,plus 8 bytes of the stack.
STRING VARIABLES in the number of characters in the variable name including the dollar sign,plus stack memory the size of the string plus one.

To store a screen row, column and character value in three numbers takes 12 bytes, in three numeric variables of one letter each, 3 bytes plus stack space of 24 bytes.

Using numeric variables, the stack space is only occupied once for each variable, so if the numbers are used frequently, use of variables may save memory.

Where a large number of such data is to be stored, memory may be saved by compacting each set into a single number:
say Row 12, Column 23, Character 32
can become a single number 122332.

To break the number up if it is in variable A,we can use:

```
ROW=VAL(SEG$(STR$(A),1,2))
COL=VAL(SEG$(STR$(A),3,2))
CH=VAL(SEG$(STR$(A),5,2))
```

Instead of using a number, it is also possible to place the data straight into a string, say A$, where A$=STR$(A).

Using a string has an advantage in that it may be up to 255 bytes long, and one string can thus contain a great deal of information.

In an adventure program for instance, 255 bytes can easily contain screen information, pointers to standard text, strength and agility counters and so on.

The only drawback is that the greater the degree of compaction, the slower your program will run.

However if your program is impossible without compaction, it is well worth looking at ways in which data can be placed into the computer using as little memory as possible.

An understanding of how the computer uses its memory is vital for this to be effective.

HOTLINE NUMBERS

The executive has expressed a desire to assist all members should you have some problems or questions, want to do some library swapping or borrow a book. This will be the place to look. Listed here are the members of the executive, committee heads, and others in the group willing to help in their specialized areas. Of course, if you wish to be placed on the list, just give me a call. I know there is a lot of expertise within our Group, so I hope to add to this list. Please respect normal hours unless you specifically know that someone doesn't mind a call at 3am, or use the BBS to leave a message at 738-0617, 24 hours a day, 7 days a week.

MICHAEL TAYLOR	PRESIDENT	831-0143
JANE LAFLAMME	VICE-PRESIDENT	(H) 837-1719 or (W) 745-2225
PETER ARPIN	TREASURER AND SYSOP	523-0017
JOHN O'CONNOR	SECRETARY	833-2626
BILL SPONCHIA	PAST PRESIDENT, SOFTWARE CONTEST	523-0878
RUTH O'NEILL	NEWSLETTER EDITOR	234-8050
TONY HOPKINS	ADVERTISING	746-4463
DAVE MORRISON	LIBRARY CHAIRMAN	737-4889
JACK McALLISTER	CASSETTE LIBRARY	225-6989
HENRI MONAT	ARCHIVES	824-0941
LUCIE DORAIS	MEMBERSHIPS	232-0393
BOB BOONE	HARDWARE/SOFTWARE	623-7841*
ART GREEN	ASSEMBLY HELP	837-1955
DICK PICHÉ	TECH	521-8667
CLUB BBS	SET MODEM TO 8N1	738-0617

* Just until mid-July, though. Until the September newsletter comes out with Bob's new number, after he moves you can contact him through his sister in North Bay at (705) 476-0265. If you like, you can also reach him on Delphi (CDU) or on Compuserve (73657,3617).

Canada 57

OTTAWA, ONT

FROM

P.O. BOX 2144, STATION D, OTTAWA
*** ONTARIO, CANADA K1P 5W3 ***

EDMONTON 99er USER'S GROUP
P.O. BOX 11983
EDMONTON, ALBERTA
T5J 3L1