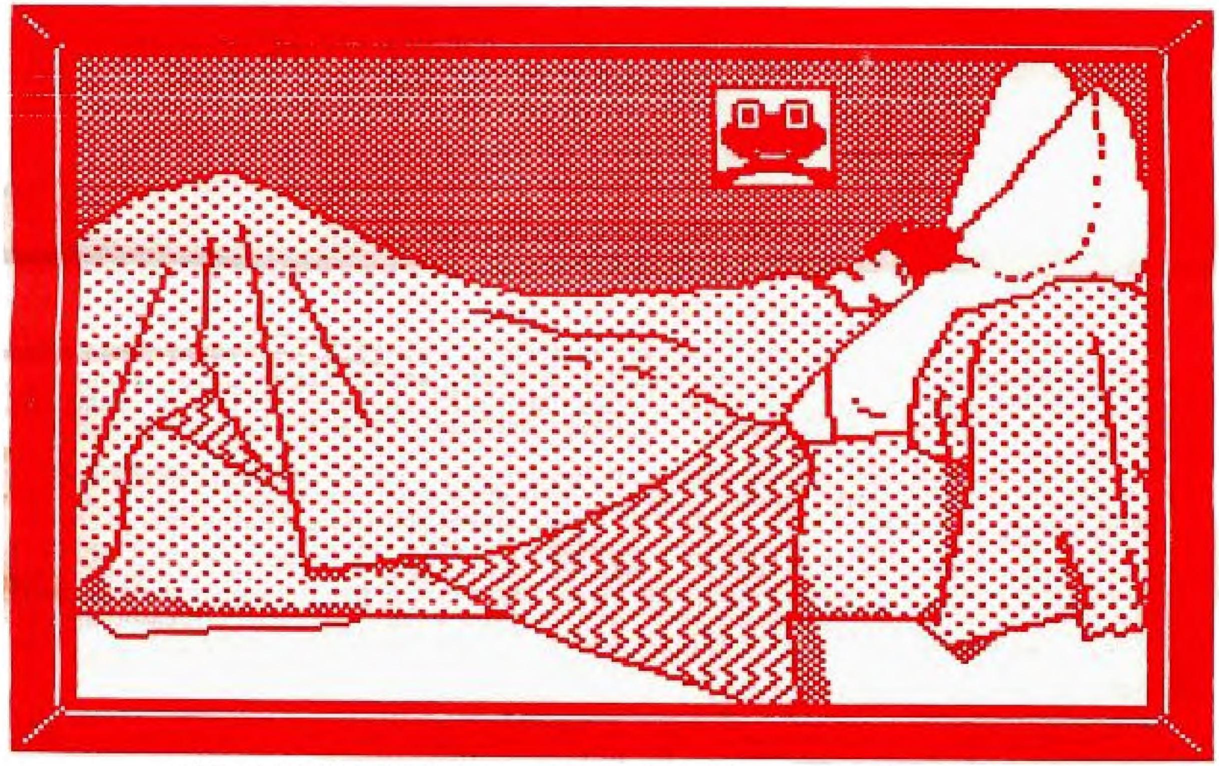
Ø65
8905

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

NEWSLETTER

VOLUME 8 NUMBER 5..........May 1989



AFTER THE FEST

DON'T FORGET THE MEETING -- May 16, 1989

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

COMING EVENTS

May Meeting:	May 16, 1989 7:30 p.m.	Merivale High School This one's in the lecture theatre downstairs.
Saturday Workshop:	May 20, 1989	No topic yet... Any ideas? Call Bill Sponchia. He'll be thrilled!
June Meeting:	June 6, 1989 7:30 p.m.	Merivale High School Back in the cafeteria, and we'll have a special guest from the RCMP! Bring a friend!
Newsletter Deadline:	May 23, 1989	(or any time before that!)
5th Annual TI-FEST	????????????	Why not? This year's was great! Let's hear a round off applause for Jane, and everyone who helped her!

EDITOR'S NOTES
from
Ruth O'Neill

I hope we haven't worried you by putting this newsletter out later than usual. Local club members are aware that the May meeting was postponed for 2 weeks, so I thought that was a great opportunity to hold the newsletter so Jane could report on the FEST while it was still news. Anyway, here it is!

I've been thinking a lot about cassette users lately... largely because of Art Green's article this month about loading assembly programs from cassette. Lucie also mentioned cassette users in her column, pointing out that her program is useful for them, too. TI users groups in general are struggling with the problem of reaching cassette users, since they are actually a much larger percentage of the TI community than disk users. Once you have a disk drive, though, it's so easy to focus on what you can do with it. The Ottawa group IS trying... especially Jack McAllister. His workshop for cassette users went well, I hear, and of course he is always ready to prepare cassettes of programs from our library. Unfortunately, I don't think he gets many requests for them. Are there any cassette users reading this? Are any of you in the Ottawa area? If so, please let us know what we can do for you. We don't MEAN to leave you out. Sometimes we need a reminder, though.

Unfortunately, I'm afraid that most of the cassette users aren't in touch with other TIers. Does anybody have any ideas at all about how to reach these people? If so, share your ideas, and maybe we can act on them before it's too late. We NEED those cassette users, and if we can show them enough support, maybe they'll find out that they need us, too.

The display at Billings Bridge was a success, I hear. Thanks, Marcelle, and everyone who helped her!

The newsletter has a variety of authors this month... thanks to all the contributors!

The President's Two Cents' Worth
by Jane Laflamme

The Fourth Annual Canadian TI-FEST is now relegated to the history books and some of us are probably glad to get back to our regular routines, and some of us wish it could have lasted longer! It was great to meet old and new friends alike. In the general admittance category, we had users from across Canada and many from the United States. It would be impossible to mention all of them and I would be afraid of missing someone, but I would like to mention two that were recently featured in two magazines. Bruce Harrison of Harrison Software, whose software was reviewed in the May issue of Computer Shopper, had a booth, and a great surprise to me, Rob Ekl, age 14, & his father with two other Pennsylvania Users arrived with the portable computer that Rob, with the help of his father, had built. You can read more of Rob in the March issue of MICROpendium.

We ended up with two more booths than were anticipated. Rob had one showing off his computer, and Peterborough's Kawartha 99ers also arrived and asked for a booth on the day. Had we known beforehand, we probably would have spaced the booths differently; we had to do some quick thinking!

Attendance was down this year, but those that arrived supported our suppliers. All of the vendors that I spoke to were quite happy with their sales for the day. Over $2500 changed hands at the used equipment sale alone!

The following companies, user groups, and inidividuals had booths: (Going clockwise around the room...)

Myarc, Inc,
Asgard Software
Harrison Software
CaDD Electronics
9-T-9 Users Group, Toronto
Garry Bowser, TASS
Computer Download Unlimited
Rob Ekl
Delphi/TI-NET
Kawartha 99ers, Peterborough
Lynx Technical Services
Laflamme & Wrigley Wholesale
CIM 99 - Montreal
Plus the Ottawa group's booths.

The following people gave talks:

Chris Bobbitt, Asgard Software
Garry Bowser, TASS
Bruce Harrison, Harrison Software
Jim Horn, Disk Only Software and Systems Manager for COMPUserve.
Sgt. Val King, R.C.M.P.
Bud Mills, Bud Mills Services, Horizon RAM disk
Lou Phillips, Myarc, Inc.
Clint Pulley, "C" Compiler
Bryan Rice, Bryght Data
Mark Van Coppenolle, CaDD Electronics

Realizing we had a wealth of information amoung our TIers at the fair, we quickly put together a programmers forum. Attending dignitaries were Lucie Dorais, Charles Earl, Art Green, and Clint Pulley, each representing their area of expertise. Thank you all for doing it on such short notice!

Now for the hard part... hoping that I mention all who helped... Hard, because I am apprehensive about missing a name. I know there were many behind the scenes of whom I am probably unaware. Please accept our thanks on behalf of our group: (In alpha order)

Stephen Bridgett	- Arranging for the table cloths (it was more work than it sounds!)
Lucie Dorais and committee	- Used equipment, posters, and helping me with a special project.
Gillian Ferringo	- Ticket sales co-ordinator

Michel Gosselin	- Making banners
Ralph Kuhn	- Distribution of posters and co-ordinating the membership drive at Billings Bridge that hopefully brought a few more users to the fair!.
Bob Lenoy	- Making banners
Jack McAllister	- Console users only booth
Mr. & Mrs. D. Morrison	- Club Library booth
Ruth O'Neill	- Distributor mail-outs and advertising on Compuserve and DELPHI
Al Palmer	- Co-ordinating systems and volunteers for the fair and set up.
Dick Piche	- Electrical & telephone setup
Bill Sponchia	- Announcements, set-up, & general support of the fair co-ordinator.
Michael Taylor	- Texlink booth, ticket sales, & Talisman Hotel co-ordination, and general support of co-ordinator

More thanks go to all those that chipped in at set-up and strike, brought their systems, and supplied refreshments at the hospitality suite; all those who did a lot of behind-the-scenes work that I didn't know of!!! Thank you also, Ottawa, for the book "Share the Flame" and the signatures and remarks inside. That will sure be a great souvenir.

As for the "bottom-line", all the bills have not been received at time of writing, but my general feeling is that we are close to breaking even, or even making a penny or two. Right on budget in other words!

The general meeting for May is changed to May 16th in the lecture hall. We were "bumped" by the school and the above date was the first available. At that meeting, I invite you show off your new piece of software or hardware, or speak of an interesting talk. Nothing formal will be planned. It will be up to you to help us out at this meeting. At the final meeting of the year, moved back to its regular time of the month, Tuesday, June 6th, Sgt. Val King will return but his presentation will differ from his talk at the fair.

Phew! I think I have said enough -- 'till next month.....

Jane

BROWSING THE LIBRARY
--with DAVE MORRISON

I recall that in a recent column, I managed to move an Australian city right across the Commonwealth, and on one disk label, I moved the Kawartha User Group from Peterborough to North Bay! So I hang my head in shame once more for this time (March issue) I referred to a music programme written by Jim Patterson. I don't know Jim Patterson; I don't know if there is a programmer by that name; and I don't know, that if there is a programmer by the name of Jim Patterson, that he has written a music programme! My apologies to Jim PETERSON of Tigercub fame, who wrote the fantastic and FREE Tigercub Country Music! An apology is also directed to Scott Morrow (and to local members!) for describing his "NUCLEAR 99" as a game! Actually, Scott has produced a simulated operation of a CANDU steam-type nuclear reactor with four screens of information. The arrow keys can be used to control the flow of primary, secondary and emergency coolants. Full documentation is on the disk. However, no construction plans are included! I gather that the screens can be printed using TI-ARTIST.

No new software has been received recently, so I cannot say what will be offered as the May Disk-of-the-Month. I can only hope that a few new non-commercial items may appear at our April 29th TI-FEST.

See you all at the Fest!

A Look At CS1
by R. A. Green

Have you ever noticed how good it feels when you stop beating your head against a brick wall? Or how obvious a solution to a problem is once you have found it?

There has been quite a bit of interest lately on bulletin boards and in Computer Shopper about loading assembler language programs from cassette. Loading assembler language programs from cassette would be very useful for those who have 32K built into the console of an otherwise unexpanded system. The problem is that the convention for loading multi-segment programs requires that the last character of the name be incremented each time a segment is loaded. That is, a three segment program, TOMB, loaded from disk causes loading of three files: "DSK1.TOMB", "DSK1.TOMC" and "DSK1.TOMD". E/A Option 5 and TIW Option 3 both know how to do this. What happens when the program is on CS1? Both E/A and TIW proceed as usual and try to load three files: "CS1", "CS2" and "CS3".

There's the problem! Too bad. The solution is a special loader. Right? Well, I thought so, and wrote one, but it's the wrong answer.

Let's back up a minute. As we all know, and are willing to tell anyone, TI did a good job designing the 4A. One of the things they did really well is the support of I/O devices so that new ones could be added easily. One of the things that allows this is the way "device/file" names are constructed. A name is in two parts. The first part is the device name proper: "DSK1", "DSK2", "RS232/1", "RS232/2", "CS1", "CS2", and so on. The second part is a series of options separated from the device name and from each other by periods. In the name, "DSK1.TOMB", "DSK1" is the device name and "TOMB" is the option. It's only the disk ROM that considers this option to be a filename in the disk directory.

The standard way to do an I/O operation is to scan up to the first period to isolate the device name; find that device on some ROM/GROM; then call the ROM/GROM routine which will process the options and do the I/O.

What do you suppose would happen if you specified an option to a device that did not expect options?

At this point, we stop beating our head against that wall and while it still feels good we tell E/A Option 5 or TIW Option 3 to load our multi segment program from "CS1.TOMB". And it happily loads three files: "CS1.TOMB", "CS1.TOMC" and "CS1.TOMD" -- the CS1 GROM routine ignoring the option it doesn't expect.

Disk of Dinosaurs -- A Review
by Michel Gosselin

This two disk package contains three animated catoon shorts on one disk and 46 TI-Artist instances along with three TI-Artist pictures on the second disk.

Of the 46 instances there are 8 realistic dinosaurs and 8 cartoon-like dinosaurs. The remainder of the instances are alphabetic blocks. The three pictures consist of a Dinosaur Hunting License and two landscapes. The landscapes can be combined with the dinosaur instances to produce interesting pictures, which can of course be used for other purposes.

My personal opinion of this package would rate 8 on a scale of ten. The main complaint I have is that the instances seem a little out of place due to their size when combined with one of the two landscapes. I have found it necessary to crop these instances with TI-Artist prior to inserting them into the landscapes. You can, of course, make your own landscapes.

I have also converted these dinosaur instances for use with Print Wizard, which turned out very well. Over all, I think this was a worthwhile aquisition and I can recommed it to anyone who likes to fool around with graphics.

Disk of Dinosaurs is written by Ken Gilliland and distributed by Asgard Software.

A Look At Assembler Language -- Calling Subroutines
by R. A. Green

Structured programming, that is, the use of subroutines, is as important in Assembler Language as it is in the higher level languages. In Assembler Language, however, it is possible and usual to use smaller subroutines and to use them more frequently. This makes it important to use an efficient calling sequence.

There are two instructions available for subroutine calls: Branch and Link (BL) and Branch and Load Workspace Pointer (BLWP), and their corresponding return instructions, Return (RT) and Return with Workspace Pointer (RTWP).

Generally, BL is used for internal subroutine calls. Internal subroutines are assembled as part of the calling program. The BL instruction saves the return address in R11 and branches to the subroutine using the same workspace as the calling code. Note that if the subroutine uses BL to call other subroutines, R11 must be saved in some way.

Generally, BLWP is used for external subroutine calls. External subroutines are assembled separately from the calling routine. The BLWP linkage is much more elaborate than the BL linkage. A new workspace is set up, and as well as saving the return address, the workspace pointer and status registers are also saved. Note that a subroutine called with BLWP can call other subroutines with either linkage without having to save any registers. On return, RTWP restores the caller's workspace pointer and the status register. Restoring the status register, we should remember, restores the interrupt mask.

Now, let's look at the "cost" of these two methods of subroutine call. An example of BL is:

```
          BL   @ISUB        Call Subroutine
          .
          .
          .
    ISUB  ...               Subroutine, ISUB
          .
          .
          .
          RT                Return to caller
```

The storage cost is 3 words (2 for the BL, 1 for the RT) and the time cost is 5 words (3 words to fetch the instructions, 2 words to save and restore the return address). An example of BLWP linkage is:

```
          REF  XSUB         Define External XSUB
          ...
          BLWP @XSUB        Call Subroutine
          .
          .
          .
          END

          DEF  XSUB         Define entry name
    XSUB  DATA XWSP,$+2     Transfer vector
          .
          .
          .
          RTWP              Return to caller
    XWSP  BSS  32           XSUB's workspace registers
          END
```

The storage cost is 21 words (3 for the instructions, 2 for the transfer vector and 16 for the workspace) and the time cost is 11 words (3 words to fetch the instructions, 2 to fetch the transfer vector, 3 to save WP, PC and ST, and 3 to restore WP, PC and ST).

This cost analysis indicates that BLWP linkage is much more expensive than BL linkage. So why use BLWP? One reason is to reduce the third cost of a program -- that is, to reduce the effort you expend writing the program. Since a routine called by BL uses tha same workspace as the caller, the two pieces of code must coordinate their use of the registers. Sharing the registers is easy for very small subroutines, but gets very difficult if the subroutine is (size or complexity. This sharing of registers also implies that bot caller and the subroutine "know" what the other is doing, which implies the subroutine is a special purpose routine just for this program.

When using the BLWP linkage there is no sharing of workspace registers, registers used by the subroutine belong to that subroutine. This situatior suited to external general purpose subroutines that are designed to independent of the caller and to be used by several programs.

Since subroutine calling and its associated parameter passing is so importa and used so frequently it is worthy of considerable thought to reduce all thre costs associated with coding them. You should read the documentation for th CALL and RCALL macros (and their macro definitions) supplied with your favorit macro assembler.

We will continue this discussion next time when we will see that in some cases the higher cost of the BLWP linkage can be overcome.

"The Universe is not user friendly"

EZ-KEYS PLUS -- A Review
by Ralph Kuhn

EZ-KEYS PLUS by Harry Wilhelm, distributed by Asgard Software, is a package of assembly routines and Extended Basic (XB) programmes that makes life in XB easier, more colourful, and gives you utilities to use in and on your XB programmes.

A (donated) checksum utility lets you list your programme so that others can check their typing, or you can check your own.

How often have you wanted a disk catalogue in XB? One keypress to call it and one for the drive number are all you need. Don't like the character set? Change it with the utilities provided. To see a possibility, you can load FATFONT or just run the XB utility programme.

A few new keypresses are added for XB editing. You can now scroll up and down through an XB line, and edit the line number without pressing Fctn X, Fctn 4, Fctn 8 THEN backspacing.

New statements are added for more control of the XB environment. Screendump, savescreen, a full screen editor, and more are there for your use. Oh, and in case you forget to save your efforts in XB, EZ-KEYS PLUS will do it for you! The automatic save utility will save your programme so you don't lose all your work to a power bump.

The big feature of EZ-KEYS PLUS is it's "macro" capabilities. Wherever a cursor appears in XB, you can send keystrokes automatically. Think about the possibilities! You can use macros in the command mode to run a programme with a single keystroke, enter options within that programme, build your own pop-up calculator, and save the information as a disk file.

Simple macros make editing easy. Deleting lines in XB is a line by line process. EZ-KEYS PLUS does NOT change that, but if you call up the first line number and press a macro key, the lines will be deleted until you end the process by pressing ol' reliable Fctn 4 (BREAK). This aborts the macro.

The importance of macro features only appear when they are used. If you've never had the pleasure, macros save you typing, time and errors. Once you've used a macro and determined that you've done what you wanted, you need never mis-type it. Has anyone NOT EVER forgotten that closing quote or parenthesis? Using a macro, it need never happen again.

The assembly additions to XB are in an area of the computer that takes no space away from your XB programme. It can be loaded elsewhere if there are other assembly routines (included by you or Systex'ed).

The package is flexible, useful, and addictive. Get in touch with Asgard or one of your local retailers. Macros and the additional XB commands make EZ-KEYS PLUS a package you'll wonder how you did without.

TEACHING PR BASE
by BILL SPONCHIA

"HOW TO RUN"

Program: PRBASE Version 2.0

(...Continued from April)

Option 4 - used to design Reports. Up to 5 reports may be designed.

i) A listing of all reports will be shown. If a report # is used then the listing will show the name given; if a report # is not yet used it will show "(Not Used)". To set up a new report select the FIRST "(Not Used)" report number. If you wish to modify a presently setup report enter that number.

ii) The screen will now display:

Report Number (1 - 5):	n	- this shows the number you have entered
Number of Columns (80/132):	80	- regular print 80; condensed print 132
Number of Lines (1 - 10):	1	- number of lines in report you wish each record to use
Report Title: (Not Used)		- input a title to identify the report. Maximum 31 characters. This prints out on top of your report and is also shown when you select Option 4 in "CREATE" or Command R in "DATA MANAGEMENT"

iii) Printer control characters may then be set up. They must be in their "Decimal" value.

Note: If all six are not used then the last one used must be "0".

eg - 12 27 78 8 0

12	- performs a form feed
27 78 8	- sets a perforation skip of 8 lines
0	- tells program that sequence is finished

iv) A listing of all fields in the record will be displayed giving their screen starting location and the field size. You will be asked for a Log Device. After you enter your printer's name and press ENTER it will ask if you wish a printing of the screen. It is my recommendation that you get the printout for future (immediate) reference. After getting the printout it is suggested that you manually write beside each field its descriptive name.

v) A suggested outline of the report is then shown on the screen. Actually what it does is start with the first field and goes to the 16th field entering each. Most likely you will have to totally redo.

Enter the information as you want to see it on your report.

Screen Location - starting position of data you wish to use

No. of Chars - number of characters from screen display you wish to use.

Note 1: Not all characters in a defined data field need be used.

Note 2: Any character, other than the graphic ones, appearing on the screen may be used. It does not have to be included in a defined data field.

Report Line - the record information can be displayed on more than one line in a report. This wants you to indicate on which line you wish it to be on.

Column Position - indicates where on the report line you wish the information just identified to start printing.

Note: All information put down in the default suggestion and not used must be erased with zeroes.

vi) The next screen will show:

```
Enter Column Header Line
1234567890.........7890   (40 characters)
^      ^     ^            (etc)
```

The second line is just a reference line.
The "^" in the third line matches up to the Column Position for each item (ie - they mark the starting point of each part of data put on the report line). They are there for reference purposes only. You may enter any column headings that you wish; the "^" will not be displayed.

vii) By pressing ENTER the report information will be saved to your data disk for future use.

Option 5 - used to design a label. Only 1 label may be defined.

i) Upon entering you will be asked "Number of lines per label: n". Enter the number of lines you wish to appear on each label. The maximum is 10.

Note: the number of lines requested should include an adequate number to facilitate the moving from one label to the next.

ii) The next screen is a listing of all fields in the record showing their screen starting location and the field size. This is the same screen as shown for report formatting (see Option 4 - (iv)).

iii) The next screen gives a suggested makeup of the label. This is the same screen as shown for report formatting (see Option 4 - (v)).
Enter the information as you want it to appear on the label. For blank lines enter zeroes in "Screen Location" and "No. of Chars". Please note that all remaining suggested items must be erased with zeroes.

iv) By pressing ENTER the label information will be saved to your data disk for future use.

Option 6 - used to set up and store up to 5 series of printer control codes.

First you are asked to describe what the printer code sequence does. Maximum length - 32 characters.

Next you are asked to enter the series of control codes.

```
- Up to 6 control characters can be entered in a series - Note: if less
  than 6 are entered then a "0" (zero) must be entered at the end
- the control character must be in Decimal equivalent
- eg.  12 27 78 8 7 0
       12       - performs a form feed
       27 78 8  - set a peforation skip of 8 lines
        7       - sounds the printer's bell
        0       - tells program that sequence is finished
```

After your sequence is entered you will be prompted for whether you wish to save it - Press "Y"

You may continue to make control sequence entries or return to the menu by pressing FCTN 9

(to be continued...)

FAST EXTENDED BASIC

LUCIE DORAIS

A nice friend wants "a good and fast sorter". Well, fast sorters need assembly language, beyond my capacities. A good assembly routine was published in COMPUTE!'s TI Collection, vol. 2, p. 226-234, and it can easily be included in this program, replacing the Sort module in lines 310-390. My sorter uses the Shell Sort, the same we used last year for the STAMPDBASE program. It comes from COMPUTE!'s Guide to the TI99/4A, by Regena, and is the fastest I could find in XB. And my program works with cassette too!

As I am busy preparing for the FEST, I will not explain the program in great detail, but it is not too difficult to understand. As a bonus, you get my newest utility for Ti-Artist, FONTSCAN.

```
100 ! SHELL SORT / L. Dorais, Ottawa U.G. / 1983-April 1989
110 !
120 DIM N$(300),ST$(1),SV$(1) :: CALL CHAR(128,"000000FFFF",129,
    "1818181818181818") :: CALL COLOR(13,14,1) :: L$=RPT$(CHR$(128),28)
130 BF=10000 :: B$="> " :: SP$="     " :: DEV$="DSK1." :: E$=CHR$(127)
140 GOTO 150 :: B,BT,C,D,I,K,N,S,SAV,SRT,LAST$,T$ :: CALL CLEAR ::
    CALL KEY :: CALL VCHAR :: !@P-
150 ST$(0)="UNSORTED" :: ST$(1)="SORTED" :: SV$(0)="" :: SV$(1)="SAVED"
160 DISPLAY AT(1,9)ERASE ALL:"SHELL SORT":L$
170 DISPLAY AT(4,1):" 1  ENTER",STR$(BF-BT)&" ch. free": :
    " 2  SEE",STR$(N)&" entries": :" 3  SORT",ST$(SRT): :" 4  SEARCH"
180 DISPLAY AT(12,1):" 5  CORRECT": :" 6  LOAD": :" 7  SAVE",SV$(SAV): :
    " 8  PRINT": :" 9  QUIT" :: CALL VCHAR(3,15,129,19)
190 DISPLAY AT(23,1)BEEP:L$:SP$&"Press a KEY"
200 CALL KEY(5,K,S) :: IF S=0 OR K<48 OR K>57 THEN 200
210 ON K-48 GOTO 220,400,300,480,600,530,530,680,710
220 CALL CLEAR ! == enter data ==
230 IF N THEN PRINT "last entry:": :STR$(N)&B$&LAST$ :: PRINT : :
240 IF BT>=BF THEN 290 ! check for max. byte space
250 DISPLAY AT(1,1):" (C)orr  (M)enu   /"&ST$(SRT):L$
260 PRINT STR$(N+1); :: LINPUT B$:T$ :: IF T$="" THEN T$=E$
270 IF T$="M" OR T$="m" THEN 160 :: IF T$="C" OR T$="c" THEN K=99 ::
    GOTO 600
280 PRINT :: N=N+1 :: N$(N),LAST$=T$ :: BT=BT+LEN(T$)+2 :: SRT,SAV=0 ::
    GOTO 240 ! BT=bytes used
290 PRINT L$:SP$&"NO MORE SPACE!":L$ :: CALL SND(-2) :: GOTO 160
300 IF SRT THEN 190 ! == sort ==
310 DISPLAY AT(24,1):"SORTING YOUR DATA..." :: B=1
320 B=2*B :: IF B<=N THEN 320
330 B=INT(B/2) :: IF B=0 THEN 380
340 FOR I=1 TO N-B :: C=I
350 D=C+B :: IF N$(C)<=N$(D)THEN 370
360 T$=N$(C) :: N$(C)=N$(D) :: N$(D)=T$ :: C=C-B :: IF C>0 THEN 350
370 NEXT I :: GOTO 330
380 IF N$(I)=E$ THEN N=N-1 :: I=I-1 :: BT=BT-3 :: GOTO 380
390 SRT=1 :: SAV=0 :: LAST$=N$(N) :: GOTO 160
400 ! == output to screen ==
410 DISPLAY AT(24,1):SP$&"SEE FROM # 1" :: ACCEPT AT(24,17)VALIDATE(DIGIT)
    SIZE(-3)BEEP:C :: IF C>N THEN C=N
420 CALL CLEAR :: PRINT " press <ANY KEY> to continue":SP$&"  <M> for
    menu":L$: :
430 PRINT STR$(C)&B$&N$(C): : :: C=C+1 :: IF C=N+1 THEN 450
440 CALL KEY(3,K,S) :: IF S=0 THEN 440 :: IF K<>77 THEN 430 ELSE 160
450 PRINT :L$:"  END - press <M> for MENU" :: CALL SND(1500)
460 CALL KEY(3,K,S) :: IF S=0 OR K<>77 THEN 460 ELSE 160
470 ! == search ==
480 DISPLAY AT(24,1):"SEARCH:" :: ACCEPT AT(24,9)BEEP:T$ :: IF T$=""
    THEN 190
490 FOR I=1 TO N :: IF POS(N$(I),T$,1)=0 THEN 510 :: PRINT STR$(I)&B$&N$(I)
    :" > this one? (Y/N)":L$
500 CALL KEY(3,K,S) :: IF S=0 OR(K<>78 AND K<>89)THEN 500 :: IF K=89
    THEN 160
```

```
510 NEXT I :: DISPLAY AT(24,1):SP$&"NOT FOUND!" :: CALL SND(-2) :: GOTO 160
520 ! == load/save ==
530 T$=SEG$("LOADSAVE",4*(K-54)+1,4)&B$
540 CALL KEY(3,K,S) :: DISPLAY AT(24,1):T$&DEV$ :: T$=SEG$(T$,1,1)
550 ACCEPT AT(24,10)SIZE(-12)BEEP:DEV$ :: IF DEV$="" THEN 190 ELSE
    DEV$="DSK"&DEV$
560 OPEN #1:DEV$,INTERNAL,VARIABLE 255
570 IF T$="S" THEN PRINT #1:N,BT,SRT,LAST$ ELSE INPUT #1:N,BT,SRT,LAST$
580 FOR I=1 TO N :: IF T$="S" THEN PRINT #1:N$(I) ELSE INPUT #1:N$(I)
590 NEXT I :: SAV=1 :: CLOSE #1 :: GOTO 160
600 N$(N+1)=E$ ! == corrections ==
610 DISPLAY AT(23,1)BEEP:L$:"Entry # to CORR.(or 0):"
620 ACCEPT AT(24,25)VALIDATE(DIGIT)SIZE(3):C :: IF C=0 THEN IF K=99
    THEN 230 ELSE 160
630 IF C<=N THEN T$=STR$(C)&B$ :: PRINT T$&N$(C) ELSE 620
640 PRINT :"Enter new data:": : :: LINPUT T$:T$ :: IF T$="" THEN T$=E$
650 SAV=0 :: IF SRT THEN IF T$>=N$(C-1) AND T$<=N$(C+1)THEN SRT=1 ELSE SRT=0
660 PRINT :: BT=BT-LEN(N$(C))+LEN(T$) :: N$(C)=T$ :: IF C<N THEN 610 ELSE
    LAST$=T$ :: GOTO 610
670 ! == print ==
680 DISPLAY AT(24,1)BEEP:"PIO" :: ACCEPT AT(24,1)SIZE(-28):T$ :: IF T$=""
    THEN 190 ELSE OPEN #2:T$
690 FOR I=1 TO N :: IF N$(I)<>E$ THEN PRINT #2:N$(I)
700 NEXT I :: CLOSE #2 :: GOTO 190
710 CALL CLEAR :: END ! == quit ==
720 !@P+
730 SUB SND(X) :: CALL SOUND(300,X,0) :: IF X>0 THEN SUBEXIT
740 FOR DEL=1 TO 250 :: NEXT DEL :: SUBEND
```

This program sorts only from the beginning of the data, so it is very handy for quick sorts of names, numbers, etc; the shorter the data, the more *you can* enter: the array dimension of 300 is adequate for data of about 30 char.; if you need more, and your data is shorter, change the DIM in line 120. The total space available is controlled by the variable BF, enough for 10000 characters; if you don't have the Memory Expansion, change it to 9000 in line 130. The variable BT controls the amount of bytes used.

The options are: ENTER data, SEE it, and of course SORT it. You can also SEARCH by keyword, CORRECT an entry (by its entry number; if you don't remember it, either SEE the file or SEARCH for your entry), and PRINT your data. If you wish to keep your file, you can LOAD/SAVE it to disk or cassette. In lines 170-180, the commas cause the next string to be displayed at TAB(15), a nice shortcut from the XB guru.

To adapt for CASSETTE use, just replace the following lines in the above listing. Cassette files can only be FIXED 64, 128 or 192; choose the one you need (maximum length of one data entry), and modify line 560 accordingly.

```
540 CALL KEY(3,K,S) :: DISPLAY AT(24,1):T$&"CS1" :: T$=SEG$(T$,1,1)
550 ACCEPT AT(24,7)SIZE(-3)BEEP:DEV$ :: IF DEV$="" THEN 190 ! escape option
560 IF T$="S" THEN OPEN #1:DEV$,INTERNAL,OUTPUT,FIXED 64 ELSE
    OPEN #1:DEV$,INTERNAL,INPUT,FIXED 64
```

Two FLAGS are used as status in the menu screen: for data SORTED or UNSORTED (this will also show in the data entry portion), the variable SRT is set to 1 if sorted; as soon as you enter a new item, it reverts to 0; if you make a correction, either from menu or while in data entry, SRT is reset to 0 if your new data does not fit between the previous and next ones (see line 650). The flag SAV (menu screen only) will tell you if your data is saved or not, to remind you to save it before quitting; it is also reset to 0 as soon as you make a change, by entering new data, sorting, or correcting an entry (see lines 280, 390 and 650). A temporary flag K is set to 99 in line 270, when making a correction while entering data; it tells Tex to go back to data entry, not to the menu (see line 620).

CALL KEY is, I hope, put to its best use in this program. In PGMWRITER and BARBIE, we used a keyboard "3" to read lower case letters as upper case ones. But the use of 3 has its drawbacks: when a CALL KEY uses a keyboard "0", as in the usual CALL KEY(0,K,S), the "0" means: "same as last CALL KEY", so that in fact Tex is still in mode 3. Another unpleasant result is that the keyboad mode also affects the ACCEPT routine: in mode 3, it will accept only upper case characters, even if the alpha-lock key is up. I finally discovered that to reset the keyboard to its full potential, you need to use a value of "5".

To better understand CALL KEY, type the following little thing; remember that mode 1 deals with the left half of the keyboard, while 2 deals with its right side; the ASCII of a key pressed being below 20, it will not show on screen. But, unlike a value of 3, modes 1 and 2 don't affect the ACCEPT routine: the whole keyboard is available!

```
100 DISPLAY AT(1,8)ERASE ALL:"CALL KEY DEMO": :"   release the alpha-lock":
    " use SHIFT for upper case": : :"K'BRD CALL KEYS ACCEPT"
110 FOR KB=0 TO 5 :: R=2*(KB+5) :: DISPLAY AT(R,2):KB
120 DISPLAY AT(24,8)BEEP:"PRESS 3 KEYS" :: FOR C=1 TO 3
130 CALL KEY(KB,K,S) :: IF S=0 THEN 130 ELSE CALL HCHAR(R,2*(C+4)+3,K)
140 CALL SOUND(100,2000,3) :: FOR D=1 TO 50 :: NEXT D :: NEXT C
150 DISPLAY AT(24,8)BEEP:" ENTER A WORD" :: ACCEPT AT(R,20):A$ :: NEXT KB
```

My SHELL SORT program uses a CALL KEY(3) whenever we check only a few keys: the IF statement is shorter, checking only the upper case ASCII value of K. BUT: from the menu, we can go to ENTER or CORRECT, which both need lower case, so I used a CALL KEY(5) in line 200. Which brought another problem: if I LOADed or SAVEd after entering data, the ACCEPT for the filename took both cases, and I had to play with the alpha-lock. So, in line 540, just before accepting the filename, I put a dummy CALL KEY: it does nothing but set the keyboard to 3, so even if you enter your filename with the alpha-lock up, Tex will read it as all upper case.

The data is entered in line 260; if you enter nothing, T$ takes the value of E$, defined as CHR$(127). This is the same trick as in the stamp data base: when you sort your data, all "empty" entries will be at the end, and line 380, in the sorting portion, will delete them and reset the N number of entries accordingly. E$ is also used upon entering CORRECTION (line 600); by setting the next after last entry to it, line 650 will work propoerly if we corrected the last entry.

Even if LINPUT strings are limited to five screen lines, Tex's memory can handle strings as long as 255 characters. This can be handy for longer data, and especially if you use this program as a crude data base. Just decompose your input into smaller strings (don't forget to pre-scan them in line 140), then reconstruct T$. Here is an example:

```
260 PRINT STR$(N+1)&B$; :: LINPUT "NAME: ":T$ :: IF T$="" THEN T$=E$
271 LINPUT "FIRST NAME: ":FNAM$ :: LINPUT "ADDRESS: ":AD$ :: LINPUT
    "CITY/PROV.: ":CI$
272 LINPUT "POSTAL CODE: ":PC$ :: LINPUT "PHONE: ":PH$ ::
    T$=T$&", "&FNAM$&" - "&AD$&", "&CI$&", "&PC$&" ph: "&PH$
```

Now for the second program. A lot of fonts are available to TI-ARTIST and C.S.G.D fans. But if only you knew which characters were in the font, and what they looked like, without having to load the graphic program and type the whole range just to see! Peter Hoddie's FONT WRITER and GRAPHIC EXPANDER have a scan option, but perhaps you don't own them; and they have a drawback: all characters being listed together, the symbols and punctuation char. are hard to spot. So I wrote this quickie, to display the characters in four categories: upper case, lower case, digits and others (symbols and punctuation). The letter "A", if present, is drawn on the screen in actual size.

```
100 REM ** TI-ARTIST FONTSCAN / L.Dorais, Ottawa U.G. / April 1989
110 ON ERROR 430 :: CALL CHAR(142,"000000FF",143,"007E425A5A5A427E")
120 B$=RPT$(" ",8) :: L$=RPT$(CHR$(142),28) :: LD$=B$&SEG$(L$,1,12) ::
    L$=B$&SEG$(L$,1,20) :: F$="1."
130 GOTO 150 :: A$,B$,C$,DT$,F$,LC$,P$,SB$,UC$,A,C,DT,HB,K,LB,LC,P,R,RS,
    S,SB,UC,V,X,Y
140 CALL HCHAR :: CALL KEY :: CALL CHARSET :: CALL ERR !@P-
150 DISPLAY AT(6,1)ERASE ALL:"UPPERCASE":L$: :L$: :"LOWERCASE":L$: :L$: :
    LD$:"DIGITS  ":LD$: :LS$:"OTHERS":CHR$(143)&"=space":LS$
160 DISPLAY AT(1,1):"SCAN WHICH FONT?": :"DSK"&F$ :: CALL CHARSET ::
    UC$,LC$$,DT$,SB$=""
170 ACCEPT AT(3,4)SIZE(-12)BEEP:F$ :: F$="DSK"&F$
180 IF POS(F$," F",6)=0 THEN F$=F$&" F"
190 OPEN #1:F$,INPUT :: UC,LC=4 :: DT,SB=12 :: RS=21 ! display column
200 ! ** scan file **
```

```
210 LINPUT #1:B$ :: IF EOF(1)THEN 280
220 IF LEN(B$)>1 THEN 210 ELSE A=ASC(B$)! found a char. (LEN=1)
230 IF A>=65 AND A<=90 THEN CALL HC(8,UC,A,UC$) :: IF A<>65 THEN 210 ELSE
    GOSUB 350 :: GOTO 210 ! upper case; if A, draw it
240 IF A>=97 AND A<=122 THEN CALL HC(13,LC,A,LC$) :: GOTO 210 ! lower case
250 IF A>=48 AND A<=57 THEN CALL HC(17,DT,A,DT$) :: GOTO 210 ! digits
260 IF A=32 THEN A=143 ! space shows as a square on screen
270 CALL HC(RS,SB,A,SB$) :: IF SB<30 THEN 210 ELSE SB=12 :: RS=22 ::
    GOTO 210
280 CLOSE #1 :: DISPLAY AT(24,2)BEEP:"[A]NOTHER  [P]RINT  [Q]UIT" ! end
290 CALL KEY(3,K,S) :: IF S=0 OR K<>65 AND K<>81 AND K<>80 THEN 290
300 IF K=65 THEN F$=SEG$(F$,4,2) :: GOTO 150 ELSE IF K=81 THEN END
310 OPEN #1:"PIO" :: B$="    " :: C$=CHR$(14) ! print
320 PRINT #1:B$&B$&"  FONT: "&C$&SEG$(F$,6,10): :B$&"UPPER CASE:  "&UC$:
    B$&"LOWER CASE:  "&LC$:B$&B$&"DIGITS:  "&DT$:B$&B$&"OTHERS:  "&SB$
330 PRINT #1:"":"":"" :: GOTO 280
340 ! ** sub draw "A" **
350 DISPLAY AT(1,19)BEEP:"A>" :: LINPUT #1:A$ :: C=VAL(SEG$(A$,1,1)) ::
    R=VAL(SEG$(A$,3,1)) :: K=127
360 FOR X=1 TO R :: FOR Y=1 TO C :: LINPUT #1:A$ :: S=1 :: C$=""
370 P=POS(A$,",",S) :: IF P=0 THEN P=LEN(A$)+1
380 V=VAL(SEG$(A$,S,P-S)) :: IF V=0 THEN C$=C$&"00" :: GOTO 400
390 HB=INT(V/16) :: CALL CBYTE(HB,C$) :: LB=V-16*HB :: CALL CBYTE(LB,C$)
400 S=P+1 :: IF S<=LEN(A$)THEN 370
410 CALL CHAR(K,C$) :: CALL HCHAR(X,Y+23,K) :: K=K+1 :: IF K=142 THEN K=91
420 NEXT Y :: NEXT X :: RETURN
430 CALL ERR(X,Y,HB,LB) :: IF X=130 THEN ON ERROR 430 :: RETURN 170 ! if
    error is I/O: go back to accept filename
440 DISPLAY AT(24,1)BEEP:"ERROR:";X;" in LINE";LB :: STOP ! other errors
450 !@P+
460 SUB HC(R,X,A,A$) :: CALL HCHAR(R,X,A) :: IF A<>143 THEN A$=A$&CHR$(A)
    ELSE A$="sp "&A$
470 X=X+1 :: SUBEND
480 SUB CBYTE(X,C$) :: IF X<10 THEN C$=C$&CHR$(X+48)ELSE C$=C$&CHR$(X+55)
490 SUBEND
```

The DIS/VAR 80 font file is read line by line; when a sole character is encountered, it means "this is the character defined next"; line 220 will take its ASCII value A, and the sub HC will show it on the screen, in the appropriate category; it will also be added to a string for the printout. If the character is "A", the program jumps to a subroutine.

To draw "A", we linput the next line in the file: in line 320, we extract the total columns and rows; we don't need the third value in the line, "pixel jump". Each subsequent line contains the decimal equivalents of the hex bytes for one character definition, separated by commas. These values are read and transformed into their hex value by CALL CBYTE; when the string C$ is complete, we CALL CHAR a character above 127 and CALL HCHAR it in the upper corner of the screen (for very big characters, counter K reverts to 91 upon reaching char. 142, already used for the screen display.)

When all the file has been read, you can scan Another file, Print a listing (minus the graphic "A"), or (Q)uit. If you wish to get a graphic dump of the screen, with the "A", you can use an assembly screen dump at this point (CALL LOAD the file before line 150; CALL LINK replaces 310-320).

To also scan C.S.G.D. fonts, add the following lines (unfortunately, the "A" will not be displayed, as I don't know yet how it is done!). These fonts all follow the same pattern: UC only, UC/DIGITS/OTHERS (always the same ones), UC/LC/DIG/OTHERS; the value A, total number of char. in the file, tells the story. When you wish to scan a C.S.G.D. font, you MUST add the "/CH" suffix to warn Tex...

```
171 IF POS(F$,"/CH",6)>0 THEN GOSUB 423 :: GOTO 280
422 ! ** sub C.S.G.D. **
423 OPEN #1:F$,INPUT,INTERNAL,VARIABLE 254 :: INPUT #1:A,C,P,R
424 IF A<28 THEN 428 ELSE IF A>44 THEN 425 ELSE 426
425 LC$="abcdefghijklmnopqrstuvwxyz" :: DISPLAY AT(13,2):LC$
426 DT$="0123456789" :: DISPLAY AT(17,10):DT$
427 SB$="!@.,"":-" :: DISPLAY AT(21,11):SB$
428 UC$="ABCDEFGHIJKLMNOPQRSTUVWXYZ" :: DISPLAY AT(8,2):UC$
429 CALL HCHAR(21,12,143) :: SB$="sp "&SB$ :: RETURN
```

TI BASIC continued from April
by Steven Shaw

APPENDICES

KEY CODES

It is sometimes useful to enter ASCII codes from the keyboard outside the usual range - for instance when PRINTing a line of defined characters.

By switching the computer to PASCAL mode, using CALL KEY(4,A,B), the range of codes available is increased.

Although the usual function codes (eg cursor control) are deactivated in Pascal mode, upper and lower case characters are not affected.

The following table gives the codes available in PASCAL mode.
The keys to be pressed are indicated thus:
A = Key A only
FA = FCTN and A together.
CA = CTRL and A together.
SA = SHIFT and A together.

CODE:	KEYS:
0	C,
1	CA
2	CB
3	CC
4	CD
5	CE
6	CF
7	CG
8	CH
9	CI
10	CJ
11	CK
12	CL
13	CM
14	CN
15	CO
16	CP
17	CQ
18	CR
19	CS
20	CT
21	CU
22	CV
23	CW
24	CX
25	CY
26	CZ
27	C.
28	C;
29	C=
30	C8
31	C9
32	SPACE
33	S1
34	FP
35	S3
36	S4
37	S5
38	S7
39	FO
40	S9
41	S0
42	S8
43	S=
44	,
45	S/
46	.
47	/

CODE:	KEYS:
48-57	0-9
58	S;
59	;
60	S,
61	=
62	S.
63	FI
64	S2
65-90	A-Z
91	FR
92	FZ
93	FT
94	S6
95	FU
96	FC
97-122	a-z
123	FF
124	FA
125	FG
126	FW
127	FV
128	na
129	F7
130	na
131	F1
132	F2
133	na
134	F8
135	F3
136	FS
137	FD
138	FX
139	FE
140	F6
141	na
142	F5
143	F9
144-176	na
177	C1
178	C2
179	C3
180	C4
181	C5
182	C6
183	C7
184	F,
185	F.
186	F/

CODE:	KEYS:
187	C/
188	F0
189	F;
190	FB
191	FH
192	FJ
193	FK
194	FL
195	FM
196	FN
197	na
198	FY
199-	na

To use:
ENTER PASCAL MODE WITH CALL KEY(4...

AFTER THAT USE THE KEYS INDICATED AND THAT CHARACTER WILL BE PRINTED: IT MAY NOT BE VISIBLE IF YOU HAVE NOT DEFINED IT.

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.

JANE LAFLAMME	PRESIDENT	(H) 837-1719 or (W) 745-2225
AL PALMER	VICE PRESIDENT	594-9216
MARCELLE GIBSON	SECRETARY	233-2384
BILL SPONCHIA	TREASURER	523-0878
MICHAEL TAYLOR	PAST PRESIDENT	831-0143
PETER ARPIN	SYSOP	523-0017
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	(705) 476-9391
ART GREEN	ASSEMBLY HELP	837-1955
DICK PICHÉ	TECH	521-8667
CLUB BBS	SET MODEM TO 8N1	738-0617

Don't forget the Trading Post!

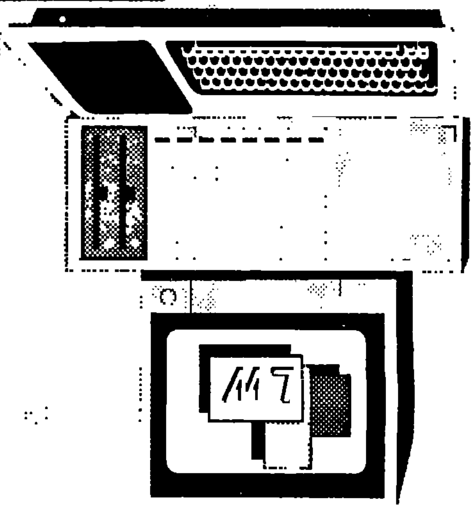
It's here for you!

Deadline to get ads into the June newsletter:

May 23, 1989

LAFLAMME & WRIGLEY
DATA (Phones):

Ottawa BBS (TEXLINK) (613) 738-0617
DELPHI "JANELAFLAMME" (No space)
CompuServe "76046,2006"



Authorized Distributor for the following:

Support your Canadian dealers:

CANARIA DATA INC.
264 Weber St. W.
Kitchener, Ontario
N2H 4A6
(519) 578-3873

COMPUTER DOWNLOAD UNLIMITED
8 Talon St.
North Bay, Ontario
P1A 1N5
(705) 476-9391
Data (TEXLINK) 476-3043
CompuServe "73657,3617"
DELPHI I.D. "CDU"

DATA*PORT
2846 Gottingen St.
Halifax, N.S.
B3K 3E1
(902) 454-0232
Data (TEXLINK) 455-2076

DEALER ENQUIRIES WELCOME

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