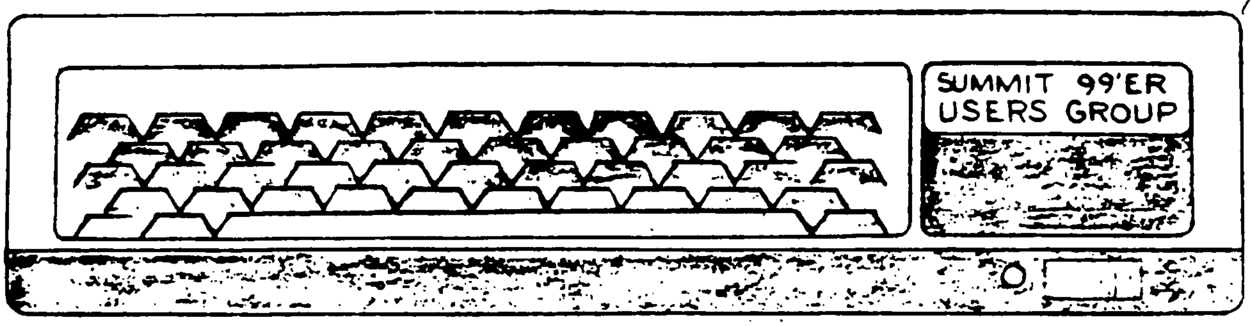
Summit U.G.
Nov 85



NOVEMBER 1985 Vol. 3 No. 11

The November meeting will be held on Thursday, Nov. 21 at Cuyahoga Falls High School at the corner of Fourth and Stow Streets in Room 413-Physic's Lab. The room will be open at 7:00 and the meeting will get under way at 7:30. The December meeting will be held Dec. 19. Please remember to sign in.

This month we will be holding elections instead of having a program and the December meeting will be our Christmas party.

There is a tape missing from the library. It is Wall Street - 9000-8. Please return it to the library.

We are looking for a new place to meet. If you know of a good place that we could meet for free, call either Mr. Figard, Mr. Hayden or Mr. Van Boven.

LIST OF BOARD MEMBERS AND THEIR HOME PHONE NUMBERS

President, Norm Sorkin	678-2360
Vice President,	
Librarian, Bert Haase	753-7846
V.P. Program,	
Secretary, Vicky Chrisman	784-0943
Treasurer, Betty Duncan	633-5217
Educational Director, Rich Williams	626-2423
Editor, Kathi Anderson	923-7530

GETTING STARTED IN PROGRAMMING BASIC

This is for those of you who have said or thought, "I'll be able to write that program as soon as I learn enough BASIC. If you have gone through the "blue book", and think you understand half of what you studied, you know enough to start that special program. Not enough to finish it but enough to start.

Don't let that last sentence throw you. Few non-professions know enough BASIC to tackle a large program and not get stuck at least once. It just means that you will have a problem to tackle and on your first program, probably more than one. More on that later.

First, learn "enough BASIC". Study the "blue book" or take Rich Williams' course in TI BASIC.

Next, get started. Getting started may seem like the hardest part of writing a program but, believe me, spending hours of work, getting half way through and then realizing that you're lost, have a mistake and don't know what to do next is much worse. Get started by getting organized.

You can't solve a problem until you know what it is, so carefully write down what problem your program will solve. I'm working on a program to catalog my color slides. The problem I am trying to solve is to be able to select slides by subject, location, or time taken.

Next, list your desired outputs, either screen or printer. Rough sketches may help to make sure you don't forget anything.

Now list the input data needed for the program.

Finally, list all the different routines that will be needed to process the input data and produce the screen or printer outputs. My slide catalog program has the following:

initialization
load data from cassette
save data to cassette
add data to file
delete data from file
input search catagory info
search file for specific records
select catagory for sorting
sort complete file
display search list
display complete file
print search list
print complete file
title
error messages
and misc. utility routines

The idea is to keep reworking and detailing your outline until you have a list of routines that can be written and tested as individual units.

What you are doing is following some of the concepts of Structured Programming. For a much better understanding than I can give here, see the series of articles in "Creative Computing" starting in May, 1984 by Arthur Luehrmann. These articles have had more influence on improving my programming than anything else I have read. They're available at the Cuyahoga Falls Library. The book "How to Build A Program" by Jack Emmerichs covers strucstructured programming and much more.

Another part of the structured concept is top down programming. The entire outline of the program is written with the branching as required for your outline. Once this is tested and working, you then proceed to write each segment and test it before going on.

By now you have probably determined that your program is a straight-line type where data is entered, worked on and displayed or it needs a menu because the user will want to jump around and use it's features in no particular order.

To help you get started, the next two columns will give listings for generic straight-line and menu programs that you can enter and modify to fit your needs.

RUSS COOK

REPRINTED FROM SAN ANTONIO AREA 99ERS NEWSLETTER, VOL 2 NO 3, AUGUST 1985

Using All 256 ASCII Characters with TI-WRITER

By David A. Aragon

In addition to writing my normal FORTH article, found elsewhere in this newsletter, I have included a short article on accessing the characters above ASCII code 127. I decided to write this article because I had a need to utilize some of the special characters on my printer (Gemini-10X) in a report I was writing. I looked through every article I could find, asked everyone I knew, and even looked in the TI-WRITER manual, but nothing. It took me a little while to figure it out, which is embarassing because it's so simple, and while I'm not sure how many of you have a need to access these codes, there might be a few of you who do, and maybe this will help you. Please remember that the procedure I give is for the Gemini printer. Some modification may be necessary for other printers.

If you look on page 240 of your Gemini User's Manual you will see a chart that looks something like this:

DEC		0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
	HEX	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	NUL		SP	0	@	P	`	p	NUL		┘	Ť	Ā	¥	SP	┌
1	1		DC1	!	1	A	Q	a	q		DC1	┐	Å	â	Ä	▘	─
2	2		DC2	"	2	B	R	b	r		DC2	╰	♣	ç	ö	▖	┐
3	3		DC3	#	3	C	S	c	s		DC3	╭	θ	£	Ü	▝	┬
4	4		DC4	$	4	D	T	d	t		DC4	↑	è	ã	¢	▗	├
5	5			%	5	E	U	e	u			↓	₧	µ	Ñ	▚	│
6	6			&	6	F	V	f	v			←	Ω	▘	ä	▞	└
7	7	BEL		'	7	G	W	g	w	BEL		→	ü	,	ö	▀	┘
8	8	BS		(	8	H	X	h	x	BS		○	Σ	†	u	▄	┴
9	9	HT		)	9	I	Y	i	y	HT		▲	σ	§	ρ	▌	┤
10	A	LF		*	:	J	Z	j	z	LF		▼	∞	▉	Ö	▐	┼
11	B	VT	ESC	+	;	K	[	k	{	VT	ESC	▸	π	◘	é	▛	◤
12	C	FF		,	<	L	\	l	|	FF		◂	±	ų	ú	▜	◢
13	D	CR		-	=	M	]	m	}	CR		◇	Ɔ	R	è	▙	◥
14	E	SO	RS	.	>	N	^	n	~	SO	RS	◆	×	½	₣	▟	◣
15	F	SI		/	?	O	_	o	DEL	SI		□	+	‖	f	█	

Looking at the above chart several things should be noted. First, by using this chart you can quickly determine the ASCII value for any particular character you need. You will notice too that all the characters available from the keyboard are used so it was impossible to use the .TL command to transliterate any of them. Now, if you look even closer you will notice that there are several codes that the printer ignores, specifically ASCII codes 1 thru 6, 16, 21 thru 26, 28, 29, and 31. Lastly, you should notice that the codes from 128 thru 159 do the same thing as the codes from 0 thru 31. This might lead you to wonder if the codes from 160 thru 255 are the same as the codes from 32 thru 127. Well, they are...almost, the difference being that above ASCII code 127 the eighth bit is 1 instead of 0. You can see this by comparing the binary code for character 63 (00111111) and the binary code for character 191 (10111111) for example.

By using .TL and a few of the unused lower codes it became simply a matter of redefining that character to tell the printer what I wanted it to do. For instance, ".TL 26:32,27,62" tells the printer to print a space and then do an ESC >. This tells the printer to process all subsequent data with the eighth bit regarded as 1. On the other hand, ".TL 28:27,61,32" tells the printer to print a space after doing an ESC =, which is the opposite of the ESC > in that all subsequent data is processed with the eighth bit regarded as a 0 (see pg. 139 Gemini users guide). Now, to call them into use, press CTRL U SHIFT Z to go into the special character or graphics mode and CTRL U FCTN Z to get back out of it (see pg 146 TI-WRITER manual). What goes in between them would be the letter(s) below ASCII code 127 that correspond to the special character(s) above ASCII code 127 that you want to include in your document. The space character is not necessary for this to work, but I recommend it to make things a little easier. The character generated takes up a space on the screen but is not printed. By putting the space character (32) in the .TL definition it makes it much easier to line things up to look right. It now becomes much simpler to print formulas or even include simple graphics in your document.

For example:

$$X^2 = \frac{\Sigma \; (\; x \; - \; \bar{X} \;)^2}{\bar{X}}$$

And, just so you don't think that I have completely forgotten about FORTH, one method to use the characters above ASCII code 127 while in a FORTH environment is to say, for example, ": 8BIT 32 EMIT 27 EMIT 62 EMIT ; and : 7BIT 27 EMIT 61 EMIT 32 EMIT ;". This is used the same way as I described above, or just say 191 EMIT and it will print a + sign if you have set the dip switches to accept 8 bit data (see pg. 52 fig. 4-7 Gemini users manual).

Well, that's it for this time. See you all at the next meeting.

This article comes to us from TI Riverside Users Group, Sept. 1985 issue.

TI-WRITER: Using the Special Character Mode

The Special Character mode of the TI-WRITER software is a method of sending the necessary commands to a printer in order to activate the many functions, such as different character fonts. Those who are familiar with using the Text Formatter of TI-WRITER may already know of using the Tansliterate command to do just this. With Special Character mode, the TL commands are not used. Instead, a number of "Special" characters, other than the normal ASCII range of 32 (space) to 127 (DEL), are generated and sent to the printer upon printout with either the PF command or the Text Formatter.

We have all seen the symbol that represents a Carriage Return installed at the end of a sentence or paragraph in our document, this is one of those "Sp-Ch"'s. The ASCII value of a Carriage Return is 13, and to send a CR to a printer you must send character number 13 (Not the value 13).

To activate the Special Character mode, you hit a CTRL-U, and the cursor symbol changes to an underline character. A second CTRL-U puts you back to normal mode.

To install a Sp-Ch in your document, you must first know what function of your printer you wish to invoke. Most printer manuals have a chart that lists the various funtions and the codes needed to activate them, and it is very handy to have a copy of this list nearby when formatting a document. For instance, the sequence ESCcape-"E" (ASCII values 27 & 69) will invoke Emphasized print on Epson printers, and if we send an ASCI value of 27, then an ASCII value of 69, the printer switches to Emphasized print. We know that ASCII-69 is a capital E, but ASCII-27 (ESCape) is not a "typeable" character. Now we go to the list of Special Characters on page 146 of the TI-WRITER manual, and we see that ASCII-27 can be generated by typing the FTCN-R while in Sp Ch mode. The symbols generated by each Sp Ch are also listed, and almost all are the Hexadecimal value of the ASCII code, compressed down to take up the space of only one character. ASCII-27 is HEX-1B and you will see a little "1b". Right after the "ESC" character, you type the "E" for ASCII-69. In summary, the sequence to send the control codes for Emphasized print would be:

```
CTRL-U
   (FTCN-R)E
CTRL-U
```

These Special Characters can be installed anywhere in the text, as they do not print upon output, just as the "re-defined" characters used with the Transliterate command are "invisible."

There are pluses and minuses to using the Special Character mode against using the Transliterate command. The TL commands are more versatile, and can easily be made to send a complicated sequence of ASCII values, where using Sp-Ch mode would get quite tedious each time a lengthy code was sent. Also, a number of TL commands can be stored in a separate file, and linked to the document upon printing, thus saving having to re-write them each time. On the other hand, when a relatively short code sequence is needed, Sp-Ch is much simpler, and the biggest advantage is that you need not load and run the Text Formatter, which can be a major obstacle to many.

As an example, when I want to just write a little note, and I want it in Emphasized, I can simply start with a CTRL-U/FTCN-R/"E"/CTRL-U and when I use the PF command, I have a nice dark print. Another widely used area is when re-formatting paragraphs, such as when modifying margins, and you need to install a Carriage-Return symbol at the end. One way is to move the cursor to the point where you need the symbol , hit CTRL-8 (New Paragraph) and then edit out the extra line and spaces. A much simpler way is to just locate the cursor and hit CTRL-U/"M"/CTRL-U, which will generate a CR symbol.

-From the "TI Forum" on COMPUSERVE ### as printed in the Ninety-Niners of the Vancouver Area Newsletter "VANEWS #24"

NOTE: Both Transliterate and Special Character mode may be used in the same document that is to be printed through the FORMATTER. -Steve Shouse, TI RUG Newsletter Editor.

TIMP TIPS AND TECHNIQUES

Steve Zimmerman

This time around, I'm going to talk (or write!) about a variety of small and not-so-small tips to help you avoid trouble with your spreadsheet models. First, if you choose to lock the formulas in a worksheet make an unlocked backup BEFORE YOU LOCK THEM! Locking the formulas in a worksheet is easy-unlocking them requires you to do it piece-by-piece! So, just in case you need to make changes in the future, keep your unlocked backup available! If someone else will be doing data entry on a complex worksheet, it IS a good idea to have them working with a locked copy- this avoids problems such as having someone enter a number or label in a cell which contains formulas or information you use elsewhere in the worksheet.

When building a worksheet, work on one area at a time. This allows you to enter numbers to check to see that each small area of the worksheet does what you want it to do. Using this technique, you can build up each individual area so that it works, and then link the areas to produce subtotals, grand totals, and the like. This is rather like programming in FORTH -in that you define and test a word, and then use that word in further definitions once you have tested it.

Deleting your numbers can help you see if an error condition results, and you can then correct the cause of the error condition in setting up your formulas.

Use relative references wherever possible in building a worksheet (I seldom use absolute references for anything!). The greatest advantage of relative references is that they allow you to easily copy formulas which will be used over and over. An absolute reference cannot be copied and used in another area of the worksheet without being edited-a time consuming process, and one which is prone to errors!

I use relative references for adding up columns of numbers--say, to get a daily total. Adding up R[-1]C+R[-3]C+R[-5]C adds the number 1 row up, 3 rows up, and 5 rows up, in THIS column (and so, can be copied to any column and do the same thing!), and places the total here. A formula of R13C2+R11C2+R9C2, will add the values in those 3 cells and place the total in whatever cell it is in. If copied from column 2 to column 3, it will STILL add up the numbers IN COLUMN 2--NOT IN COLUMN 3! To get it to add up the numbers in column 3, 4, 5, or wherever it is copied to, it must be edited to correct the column references after it is copied. This is, to put it mildly, a pain!

The most accurate way to enter formulas in a spreadsheet is by pointing. Some spreadsheets don't allow this, but we are fortunate because mulTIplan does. In fact, in mulTIplan, we can point not only in entering but also in editing formulas. This is sometimes referred to as "wander mode". Using this method of building a formula is simple--and also has the advantage of creating relative references.

To use this method, begin by placing the cursor or cell pointer in the cell in which you want the formula to be placed and the end result to appear. Key either V or = to begin entering a formula. Use the (fctn) E, S, D, or X (the arrow keys) to move the cursor to the first cell you want to add, subtract or whatever. As you do, you will see a relative reference appear on the command line after the word VALUE:--if you move the cell pointer up 1 cell, it will show VALUE: R[-1]C. If this is the first cell you want to use, press the key for the operator (+,-,*,/) you need (in this case, let's say +). The + will appear after the cell reference on the command line and the cell pointer will drop back to the cell you are building the formula in. To select the cell you want to add to the one just above, move the cell pointer again using the arrow keys. As you move it, you will see the relative reference on the command line change. When you have found the next cell, again, enter the next operator, and continue in this manner until the formula is complete. When you have pointed to the last cell to be in the formula, press <enter>. The cell pointer will drop back to the cell you are working in, and the number created by your formula will appear in that cell. In the lower left-hand corner of your screen, you will see the coordinates of the cell you are in, and the first 16 positions of the formula you have created (or the entire formula, if it is less than 16 positions long).

Using this method, you can easily enter long formulas without trying to remember cell coordinates and without taking the time (and memory overhead!) involved in naming cells. Formulas that operate in one column or row can be copied across or down and will work properly in any row or column.

When building a model this way, enter your row or column labels first, then enter data (sample data, if possible), and last, point to each data item to create your formula. Once you have the formula set up, blank the cells with your sample data items, let the sheet recalculate (or tell it to recalculate, if you turn off the automatic recalc as I always do), and check for errors, such as #DIV/0! (meaning that you're trying to divide by 0). Save your backup copy when your formulas are done, then lock the formulas in your working copy (if you want to), and enter your data. This will help you in creating error-free models.

Next month, we'll go over some (potential) problems that can lead to your having different numbers appear in your models than you are expecting! Until then, have fun with mulTIplan!

Reprinted from THE COMPUTER VOICE
Newsletter of the Southern California
Computer Group
October 1985

TI 99/4A

This article comes to us from Boise Users Group, 10/85 issue.

FREEWARE & FAIRWARE

We now have the following software available, for anyone who wants a copy. All that we ask is if you find the program useful, send the author a reasonible amount for his work to support your computer.

Titles:

Mr Bruce Caron (Disk Manager 1000, Version II)
C/O The Ottawa TI-99/4A User's Group
P.O. Box 2144
Station "D"
Ottawa, Ont.
K1P 5W3

Mr Paul Charlton (Fast-Term, Version 1.13)
1110 Pinehurst Ct.
Charlottesville, VA 22901

Mr Tim Curran (Fast-Loading FORTH)
4153 Four Pole Road
Huntington, WV 25701

Mr Todd Kaplin (Disk Master, Version 1.6)
5802 N. Western Apt. 35
Chicago, IL 60659

Mr Tom Knight (TK-Writer & Super Disk Dupicator)
7266 Bunion Dr.
Jacksonville, FL 32222

Mr Marty Kroll (A/L Dissembler)
218 Kaplan Ave.
Pittsburgh, PA 15227

Mr Stephen Lawless (Mass-Copy Version 3.25 & 128K Writer)
2514 Maple Ave. Cedars
Wilmington, DE 19808

Mr Bob Lawson (Household Budget Management Print)
16223 Mill Point Drive
Houston, TX 77059

Mr Danny Micael (Screendump & Neatlist)
Route 9, Box 460
Florance, AL 35630

Mr Ron Rutledge (The Director)
1020 3rd Street
Waukee, IA 50263

Mr John Taylor (Sprite Builder)
2170 Estaline Dr.
Florence, AL 35630

SUMMIT 99ers USERS GROUP
P.O.Box 3201
Cuyahoga Falls, Ohio 44223

P.O. BOX 11983 EDMONTON