THE GUILFORD 99'ER NEWSLETTER

VOL.6 NO.4 APRIL 1989

Scott Hughes, Pres.(227-2688) Emmett Hughes, Vice Pres.(584-5108)
Mack Jones, Secy/Treas(288-4280) Herman Geschwind,Pgm/Library(288-0015)
BBS: (919)274-5760(OPUS) ROS (919)-621-2623

The Guilford 99'er Users' Group Newsletter is free to dues paying members (One copy per family, please). Dues are $12.00 per family, per year. Send check to 3202 Canterbury Dr., Greensboro, NC 27408. The Software Library is for dues paying members only. (George Von Seth, Editor)

OUR NEXT MEETING

DATE: April 4,1989. TIME: 7:30 PM PLACE: Glenwood Recreation Center 2010 S. Chapman Street.

Program: Dan Post will demo the McInker to re-ink your printer ribbons. A "brand new" ribbon for about a nickel!!

MINUTES

The March 7th meeting of the Guilford 99er Users' Group was held at the Glenwood Recreation Center in Greensboro, N.C. There were 4 members and 1 visitor present due to inclimate weather.

The meeting was called to order by the Secretary at 7:45 P.M. as no other members were likely to show up. The minutes were read and accepted as read.

Old Business: There was no old business discussed.

New Business: There was no new business discussed.

Since there was no business conducted, Bob Carmany gave a demo on a new program he has written. Bob also demoed a few other programs and the meeting was adjourned around 9:30 P.M.

Respectfully submitted,
L.F. "Mac" Jones, Sect./Treas.
Guilford 99er Users' Group

RAMBYTES

By "Mac" Jones

Five brave souls braved the slick streets and made it to the mee Tuesday night of the 7th. Bob Carmany, Andy Small, Tony Kleen, and t Secretary made it and as far as I know, made it back! We had Wayne Gross, a PC user, to come by since he had seen notice of our meeting. He seem to enjoy the fellowship even if we were not compatable.

Please don't feel bad for not making the trip, because to tell you the truth, if it hadn't been for my having to provide the P-box, I sure would have sat by the fire and snoozed! It was indeed a rough evening, what with the freezing rain and the freezing cold wind.

I was very pleased with Bob's new program which he demoed at the meet. I will just say that, if one programs quite a bit, it can be a big help to one. It is a program that enters many commands for you so that you don't have to do it yourself. I will let Bub tell you about it himself later.

This "cabin-fever" has been telling on me. I have been at the console for quite a few "snowed-in" days, but it looks like walking weather has finally gotten here at last. One good thing about the bad days, the Carolina Model Railroaders' of which I have been a member since 1972, had over 200 labels to be printed for a mailing of an auction/open house in April. I volunteered to do the job and it was very much appreciated by the club. I used a label program called "Name-it". I tried several times to use the software I have used for our computer group listing, but it kept crashing after about 150 names. I finally got fed-up with re-typing those over and over and used Name-It. The one thing I don't like about Name-It is that you can't delete any duplicated names when you run into them. For instance, after the mailing, we got about 5 "return to senders" that the P.O. would not deliver since the fowarding time had expired. They put the new address on the envelope, but I had no way to go into the program and make the correction. This means I had to re-enter the information and just remember to pull off the old address when it wa printed again. There are only a few of us that have computers in t club, so not everyone could understand the problem. Oh well, maybe some one will come along with a better label program soon. That's a good bad weather project I might try one day.

I left a message on the BBS to Charlie Mendenhall who is a "Big Blue" user and asked him for the info on the PC meetings as we get visitors at times wanting to know. For all's information, the PC club meets the 2nd Tuesday of the month at the King Hall at Guilford College.

Seems like everyone I talk to is having trouble with the new format of the Opus BBS. The only reason that I used it really was because it is the only one that has the TI-ECHO that unites all the users across the country. Like I mentioned, now with the new software, it's hard to log on. We are hoping that Dan Post will be able to carry the echo and then we can have all of the TI on one board. As it stands, Dan's Groundstar is the only board to carry TI files. As most of you know, Herman is co-SYsop so that is one reason we have so many new files along. Herm is a habitual user of Genie and Compuserve, so he imparts with us what he finds there.

Here's hoping to see all of you at the April 4th meeting, so until then, enjoy the good TImes.

TECH TALK

(The following came from the Chicago U.G. "S.I.G." on Delphi)

By Mike Maksimik

Some of you may have followed TI's developments in the time that the 99/4A was at it's childhood. All sorts of plans, marvels, new things for the home computer that "was ahead of it's time." There were several peripherals developed by TI but were only released in tiny quantities, mostly to the TI employees that got the pick of the crop. Some of these never made it to the production lines, but only a few prototypes survived.

The modem card, which essentially was a Novation Cat 300 baud modem, was placed on a peripheral card, and a DSR ROM was given it to control very low-level functions, such as modem-to-vdp RAM interrupt routine, powerup routine, etc. It would work with a command module, like TE II just as the disk manager module works with the low-level routines in the disk controller to perform the DOS functions. Only a very few of these survived. Another little known card was the IEEE 488 bus controller card. It contained the TMS9914 GPID (general purpose interface bus) that allowed the lab and mechanical equipment that used GPIB to interface to the TI. One could access the GPIB like a file device. This same standard is found in unexpected places. Any of you have a commodore 64? The communication bus used to connect it's ring-style bus of peripherals is a modified GPIB one of commodore's own design. The SCSI interface (small computer system interface) is essentially a multi-GPIB, allowing very fast buffered serial transfer between storage devices. SCSI also has interrupt lines to alert the host that data is waiting to be read or written. The VCR controller, a $500.00 range peripheral, along with support software, was introduced as a means to combine video from a VCR and the video from a TI. The card would control playback, hold, framing, and other functions. Digital Research created a similar product to control videodiscs that attached to an apple or a commodore 64, although much later than TI's development. The debugger card, a little known device, was in existence when the 99/4A was born. In fact, it's design can be rooted to the support hardware in the 990 minicomputer series. Essentially, the TMS9900 is a minicomputer on a chip. The editor/assembler GROM was a virtual image of the DX10 assembler used on the 990 minicomupter. Some directives one would only find on a minicomputer exist in the editor/assembler package, but were dormant in the 99/4A. The debugger board was designed to bring the 99/4A closer to a minicomputer's environment. The DEBUG program, included with the editor/assembler package, has several features that cannot be used without this piece of hardware. In fact, the editor/assembler looks as if it was taken direct from a 990 itself. The only added features were the GROM utilities, sucha VMBW, DSRLNK, LOADER, etc. that didn't support the features that a 990 could handle. It's too bad that TI wishes to keep the plans for this card on ice, it would be a dream to program with. It allowed multiple breakpoints by using the XOP 3 opcode, which would allow you to step your program through and look for errors or miscalculations. Although we can do this through software, the debugger board used a hardware approach. The design of this board, and what it contained, are up for grabs. If anybody knows, I'd appreciate you sharing with the rest of us. Send me a letter.

Still another rare peripheral was the GROM library peripheral. It essentially was a super-widget that could access ALL of the GROM in the cartridges. This would be handy for TI BASIC, since TI BASIC searches external GROM for subprograms. TI extended BASIC does this too, but doesn't search DSR ROM when a program is running. Modules like TE II, personal record keeping, and extended BASIC could all be plugged in and the CALL routines could be accessible to BASIC. BASIC could use the commands it wished to whatever, and all you had to do is plug your favorite "flavor" modules into the library peripheral to get the necessary language expansion. Imagine a GROM cartridge giving advanced graphics to TI BASIC, another for print spooling, still another for expansion memory control. Others for high speed cassette routines, etc. so the lang could expand by adding cartridges. It's the same technique used with t peripherals: the computer never becomes obsolete, because it automatically responds to any new device attached. This is true of the library peripheral. This is another device I would LOVE to see.

Some of us have the HEX-BUS controller. In the days of the 99/2, the CC40, and the 99/8, the hex-bus controller was introduced for the 99/4A t allow compatibility with these devices. Essentially, they were designe like the commodore 64's peripheral system, where a slow serial transfer was appropriate for the hex-bus devices, a disk drive wouldn't be feasible. So TI never considered the HEX-BUS disk drive. The Wafertape drive, the CAT modem, the RS232/parallel interface, and the 4-color printer, were all developed. All were battery operated and could fit in a briefcase, as did the CC40. For the 99/4A, it was an inexpensive means to expand.The hex-bus controller was a small device containing a DSR ROM that controlled the I/O drivers which "spoke" to the hex-bus peripherals. Since the main use was for the CC40, it wasn't pushed for the 99/4A. The 99/8 could also rely on the PE BOX for it's devices. It had it's own special FLEX CABLE card, which used some special control lines to expand it's own capabilities. Since the 99/8 used a TMS9995, the same as the GENEVE, it could use the extra 3 address lines in the PE BOX, giving a total address space of 2 to the 19th power, or 512 k of directly addressable memory. Since some of these banks were probably switched, the address space grew to a total of 4096 k, which is sufficient for MOST of my needs. The speed of this processor was greater, and it's throughput was even greater, but more on that later. Some other control lines were used, some to indicate a 9900 or a 9995 present in the system, some to allow multi-level interrupts, still others to initiate HOLD sequences, which are found on the mainframes, and large multi-user systems as a wa to deal with wasteful processing, and interrupt idling.

TI had a HARD DISK controller in the plans, probably MYARC's, but the technical data I have is 1982. I own a rare card. Some of you may remember a company called A/D electronics, out of Sacramento, California. They produced a control card which allowed sampling of environmental data through an 8-bit analog-to-digital controller. This device allowed hookups of many items, such as temperature probes, light transducers, etc. and was mainly used as a scientific device. Some possible uses included home control, because it also contained a real-time battery backed clock. Plus, there were separate digital inputs and outputs, for switches and relays, respectively. My main use for the A/D card, FIRST ADE, is a mouse. The RADIO SHACK color mouse contains two potentiometers turned by a rolling motion of the mouse. The potentiometers, when interfaced with the ADC0809 chip, (two channels, x and y) gives me mouse control with TI ARTIST. I wrote the DSR myself, and have been using this device for about a year and a half. The MBP clock card is a similar device, although it does not contain a digital input or output array. The ADE card, however, could also switch external relays, or sample data on 16 lines (8 in, 8 out). If timing was correct, an 8-bit parallel interface was possible. I still use this card, and the clock is handy for keeping my p-system master disk up-to date.

The FORTI music card was a device which allowed one to produce sound on not one but 4 extra TMS9919 sound generators. By arranging the frequencies on the 12 music channels available, different waveforms wer possible. Now, with the FORTI, sounds even a c-64 owner could envy were possible. And, there were 4 percussion channels independent of each other. I can imagine "AXEL-F" running on this card!! And of course, we all know of the more common peripherals, the triple tech, the disk controllers, the 32k cards, the RS232 cards. Even these make our computers sophisticated enough to meet TI's long dead expectations. I also own the p-code card, and another article is devoted to THAT!

FAIRWARE NOTES

(From the Sudbury,Ontario CANADA newsletter)

o TI-WRITER V4.0...RAG SOFTWARE..ART GREEN...NOVEMBER 1988.

This is a extensive modification to TI WRITER. It removes a lot of the annoyances of the original TI WRITER and is compatable with the original.

The documentation is extensive and describes all the changes.
Here are just some of the changes.
EDITOR
o Improvements in speed of Move, Copy, Delete lines.
o QQ exits editor immediately without any other prompts.
o "CTRL ," will put you to the top of the file i.e. line #1.
o "CTRL ." will put you to the bottom of the file i.e. line E.
o A configure program that will customize the editor. Defines printer, screen colors, initialize tabs, wordwrap on or off, line # on or off, define character set.
FORMATTER
o Speed of formatter improved
o Reduction in file size which means faster loading.
o Formatter commands can be entered in upper lower & mixed case.
o Handling of asterisks improved.
o Eight new format commands have been added.
o A configure program to define your printer for the formatter.

There is also a Supercart version included.
Art Green is well known for his MACRO ASSEMBLER along with other software releases. This is fairware and they are asking $10.00

--The current version is 4.2 and is up on ROS (Ed.)

RAG SOFTWARE
R. A. GREEN
1032 CHANTENAY DR.
GLOUCESTER, ONT. CANADA
K1C 2K9

o GRAPHIC LABELER..V3.1..STEVEN MCWATTY

This is a label program that allows you to use CSGD graphics and to display them on the left hand side of the label. This latest version allows you to catalog the CSGD graphic filenames on the screen before loading them from disk.
One of the better label programs.

STEVEN MCWATTY
RR #1
KINBURN ONTARIO
KOA 2HO
CANADA

CONFESSIONS OF A SMALL-TIME USER

By Tony Kleen

First; thank you, Bob Carmany, for your interesting program on loaders, anc interesting review of your program that creates a program. These topics h e gotten me once again interested in XBasic pograms, especially those programs that can write other programs.

If you're interested in writing this kind of program, you will need t understand TI's MERGE file techniques, and what makes this technique work. Th file characteristics can be identified easily enough by the use of any Disk Manager; the pertinent items being DISPLAY and VARIABLE163. Let's start by writing a program that will display a MERGE file.

```
11 ! DSK.GAK.LOOK1
12 ! ------------- 89/03/15
13 OPEN #1:"DSK.GAK.LOOK1:M",SEQUENTIAL,DISPLAY ,INPUT ,VARIABLE 163
14 OPEN #2:"PIO"
15 FOR AE=1 TO 1 STEP 1
16 LINPUT #1:A$
17 D$=""
18 AC=LEN(A$)
19 IF AC>18 THEN AC=18
20 FOR AB=1 TO AC STEP 1
21 CALL ZEROFILL(ASC(SEG$(A$,AB,1)),3,C$)
22 B$=B$&C$&":"
23 NEXT AB
24 PRINT #2:B$
25 !
26 FOR AD=1 TO 10
27 B$="        "
28 AC=LEN(A$)
29 IF AC>(AD+18)THEN AC=(AD+18)
30 FOR AB=(3+16*AD)TO AC STEP 1
31 CALL ZEROFILL(ASC(SEG$(A$,AB,1)),3,C$)
32 B$=B$&C$&":"
33 IF AB=AC THEN PRINT #2:B$
34 NEXT AB
35 NEXT AD
36 AE=AE*EOF(1)
37 NEXT AE
38 CLOSE #1
39 CLOSE #2
40 END
41 SUB ZEROFILL(AA,AB,A$):: A$=STR$(AA):: IF LEN(A$)>AB THEN B$="*" ELSE B$="0"
42 FOR AC=LEN(A$)TO AB-1 :: A$=B$&A$ :: NEXT AC
43 SUBEND
```

What I've done is write the program, SAVE'ed DSK.GAK.LOOK1, then SAVE'ed DSK.GAK.LOOK1:M,MERGE. This will allow us to review the MERGE file of the program we're creating to look at a MERGE file. Let's look at the program.

Lines---	Purpose---
13/14	Open our files.
15/36/37	Process our input file until we hit an EOF condition.
38-40	Close our files.

16-19	1st print line setup. This line will display 18 MERGE bytes.
20-24	Build/print the 1st print line.
26-35	Build/print the remaining 10 print lines. These lines are offset 8 print columns so one can discern the beginning

a new MERGE record.

41-43   Subprogram ZEROFILL. This assures us 3 characters per byte, so that our print columns will 'line up'.

Now, let's look at the output produced by this program....

```
000:011:131:032:068:083:075:046:071:065:075:046:076:079:079:075:049:000:
000:012:131:032:045:045:045:045:045:045:045:045:045:045:045:045:045:032:
        056:057:047:048:051:047:049:053:000:
000:013:159:253:200:001:049:181:199:018:068:083:075:046:071:065:075:046:
        077:079:079:076:049:058:078:179:246:179:162:179:146:179:243:200:
        003:049:054:051:000:
000:014:159:253:200:001:050:181:199:003:080:073:079:000:
000:015:140:065:069:190:200:001:049:177:200:001:049:178:200:001:049:000:
000:016:170:253:200:001:049:181:065:036:000:
000:017:066:036:190:199:000:000:
000:018:065:067:190:213:183:065:036:182:000:
000:019:132:065:067:192:200:002:049:056:176:065:067:190:200:002:049:056:
        000:
000:020:140:065:066:190:200:001:049:177:065:067:178:200:001:049:000:
000:021:157:200:008:090:069:082:079:070:073:076:076:183:220:183:216:183:
        065:036:179:065:066:179:200:001:049:182:182:179:200:001:051:179:
        067:036:182:000:
000:022:066:036:190:066:036:184:067:036:184:199:001:058:000:
000:023:150:065:066:000:
000:024:156:253:200:001:050:181:066:036:000:
000:025:131:000:
000:026:140:065:068:190:200:001:049:177:200:002:049:048:000:
000:027:066:036:190:199:008:032:032:032:032:032:032:032:032:000:
000:028:065:067:190:213:183:065:036:182:000:
000:029:132:065:067:192:183:065:068:195:200:002:049:054:193:200:002:049:
        056:182:176:065:067:190:183:065:068:195:200:002:049:054:193:200:
        002:049:056:182:000:
000:030:140:065:066:190:183:200:001:051:193:200:002:049:054:195:065:068:
        182:177:065:067:178:200:001:049:000:
000:031:157:200:008:090:069:082:079:070:073:076:076:183:220:183:216:183:
        065:036:179:065:066:179:200:001:049:182:182:179:200:001:051:179:
        067:036:182:000:
000:032:066:036:190:066:036:184:067:036:184:199:001:058:000:
000:033:132:065:066:190:065:067:176:156:253:200:001:050:181:066:036:000:
000:034:150:065:066:000:
000:035:150:065:068:000:
000:036:065:069:190:065:069:195:202:183:200:001:049:182:000:
000:037:150:065:069:000:
000:038:160:253:200:001:049:000:
000:039:160:253:200:001:050:000:
000:040:139:000:
000:041:161:200:008:090:069:082:079:070:073:076:076:183:065:065:179:065:
        066:179:065:036:182:130:065:036:190:219:183:065:065:182:130:132:
        213:183:065:036:182:192:065:066:176:066:036:190:199:001:042:129:
        066:036:190:199:001:048:000:
000:042:140:065:067:190:213:183:065:036:182:177:065:066:194:200:001:049:
        130:065:036:190:066:036:184:065:036:130:150:065:067:000:
000:043:168:000:
55:255:
```

If you look at the 1st 2 numbers(each representing 1 byte), we've got our line numbers 11-43; plus 1 extra line number 255:255:. This last line nu the EOF indicator to the MERGE process. If you look at the last number of line (except the EOF indicator line), these lines all end with ASCII(000).

Well, we've figured out the 1st two bytes of each MERGE line, the last t s of each MERGE line, and the extra EOF indicator line. Now, what the dickens are all those numbers in-between? To find out, let's write another program that creates a MERGE file that will list all the interpretations of the allowabl ASCII values 1 through 255. I'd like to credit Bob Carmany with showing me th technique. The program, and it's results follow:

```
11 ! DSK.GAK.LOOK2
12 ! ============ 89/03/16
13 OPEN #1:"DSK.GAK.LOOK2X:M",SEQUENTIAL,VARIABLE 163,OUTPUT,DISPLAY
14 FOR AA=1 TO 255
15 PRINT #1:CHR$(000)&CHR$(AA)&CHR$(AA)&CHR$(000)
16 NEXT AA
17 PRINT #1:CHR$(255)&CHR$(255)
18 CLOSE #1
19 END
```

To get an idea of what our GAK.LOOK2X:M file looks like, let's run GAK.LOOK1 using this MERGE file as input. The first and last lines displayed are as follows:

```
000:001:001:000
000:002:002:000
000:003:003:000
000:252:252:000
000:253:253:000
000:254:254:000
000:255:255:000
255:255:
```

So now, if we MERGE this file, we will be able to see what MERGE's cross reference for each ASCII byte value is. Enter a NEW at your terminal (to clear your computer's buffer), then MERGE DSK.GAK.LOOK2X:M . After the merging is complete, just enter a LIST command to see the cross reference list. An abbreviated list follows.

```
32
33 !
34 "
etc.
```

'Scrunching' this LIST into as few lines as possible gives the following list. I apologize for this 'scrunching', but otherwise the list would be over 3 pages in length, a waste of our newsletter space.

```
32 (SPACE) 33 ! 34 " 35 # 36 $ 37 % 38 & 39 ' 40 ( 41 ) 42 * 43 + 44 , 45 - 46
62 > 63 ? 64  65 A 66 B 67 C 68 D 69 E 70 F 71 G 72 H 73 I 74 J 75 K 76 L 77 M
78 N 79 O 80 P 81 Q 82 R 83 S 84 T 85 U 86 V 87 W 88 X 89 Y 90 Z 91 [ 92 \ 93 ]
94    95 _ 96 ` 97 a 98 b 99 c 100 d 101 e 102 f 103 g 104 h 105 i 106 j 107 k
108 l 109 m 110 n 111 o 112 p 113 q 114 r 115 s 116 t 117 u 118 v 119 w 120 x
121 y 122 z 123 { 124 | 125 } 126 ~ 127 ??? 128 ??? 129 ELSE 130 :: 131 ! 132
IF 133 GO 134 GOTO 135 GOSUB 136 RETURN 137 DEF 138 DIM 139 END 140 FOR 141 LET
142 BREAK 143 UNBREAK 144 TRACE 145 UNTRACE 146 INPUT 147 DATA 148 RESTORE 149
RANDOMIZE 150 NEXT 151 READ 152 STOP 153 DELETE 154 REM 155 ON 156 PRINT 157
CALL 158 OPTION 159 OPEN 160 CLOSE 161 SUB 162 DISPLAY 163 IMAGE 164 ACCEPT 165
ERROR 166 WARNING 167 SUBEXIT 168 SUBEND 169 RUN 170 LINPUT 171 ??? 172 ??? 173
??? 174 ??? 175 ??? 176 THEN 177 TO 178 STEP 179 , 180 ; 181 : 182 ) 183 ( 1
& 185 ??? 186 OR 187 AND 188 XOR 189 NOT 190 = 191 < 192 > 193 + 194 - 195 *
196 / 197   198 ??? 199 ??? 200 ??? 201 ??? 202 EOF 203 ABS 204 ATN 205 COS 206
```

```
207 INT 208 LOG 209 SGN 210 SIN 211 SQR 212 TAN 213 LEN 214 CHR$ 215 RND
216 SEG$ 217 POS 218 VAL 219 STR$ 220 ASC 221 PI 222 REC 223 MAX 224 MIN 225
RPT$ 226 ??? 227 ??? 228 ??? 229 ??? 230 ??? 231 ??? 232 NUMERIC 233 DIGIT 234
UALPHA 235 SIZE 236 ALL 237 USING 238 BEEP 239 ERASE 240 AT 241 BASE 242 ???
243 VARIABLE 244 RELATIVE 245 INTERNAL 246 SEQUENTIAL 247 OUTPUT 248 UPDATE 249
APPEND 250 FIXED 251 PERMANENT 252 TAB 253 # 254 VALIDATE 255 ???
```

I'm going to interpret the first 3 lines of GAK.LOOK1, then I'll return in the next article to explain more MERGE techniques.

```
000:011:131:032:068:083:075:046:071:065:075:046:076:079:079:075:049:000:
------- --- -----D---S---K---.---G---A---K---.---L---O---O---K---1-
   :     :
   :     :-> "!"
   :
   :-> LINE NUMBER 11.

000:012:131:032:045:045:045:045:045:045:045:045:045:045:045:045:045:032:
        056:057:047:048:051:047:049:053:000:
   (similar to line number 11)

000:013:159:253:200:001:049:181:199:013:068:083:075:046:071:065:075:046:
        --- --- -------:-1- --- -------:-D---S---K---.---G---A---K---.--
         :   :       :       :     :
OPEN-----:   :       :       :     :
   #---------:       :       :     :
   1-----------------:       :     :
   :-------------------------:     :
".."-------------------------------:

        077:079:079:076:049:058:078:179:246:179:162:179:146:179:243:200:
        -L---O---O---K---1---:---M- -,- --- -,- --- -,- --- -,- --- ---
                                         :       :       :       :   :
SEQUENTIAL ------------------------------:       :       :       :   :
DISPLAY    --------------------------------------:       :       :   :
INPUT      ----------------------------------------------:       :   :
VARIABLE   ------------------------------------------------------:   :
   :-----------------------------------------------------------------:
   :
   :    003:049:054:051:000:
   :    --- -1---6---3-
   :     :
# -------:
```

As you can see, MERGE has some strange interpretations for 'quotes' and 'numeric strings'.
ASCII(199) means that a QUOTE has been encountered, the next byte will give the number of bytes used to represent that QUOTE.
ASCII(200) means that a NUMBER has been encountered, the next byte will give the number of bytes used to represent that NUMBER.
(Actually CHR$(199) means that the string to follow is to be enclosed in quotes and CHR$(200) means that it is not to be enclosed in quotes---a bit different than a numeric variable---Ed.)

What I plan to do next session is create a program that will compress another program into as few a number of lines as possible. I realize that such a program already exists, but it'll be more fun to do it ourselves; and more informative.

See ya' at the next meeting.