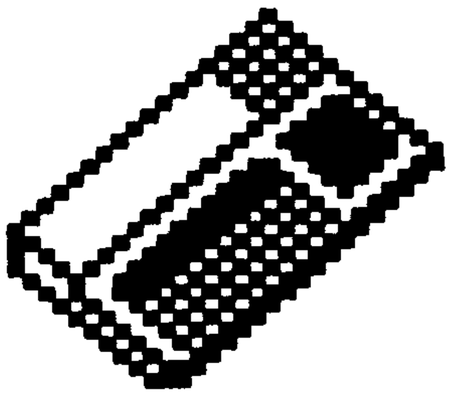

HUG

HOUSTON USERS' GROUP

FEBRUARY

1986

MEETING SCHEDULE

FIRST SUNDAY OF EVERY MONTH
(2nd Sunday if 1st Sunday
is on a holliday weekend)

HUG TIBBS - (713) 475-8909
24-hour BULLETIN BOARD

****

THE NEXT MEETING IS

SUNDAY, FEBRUARY 2, 1986 2:00 P.M.

St. John's School - 2401 Claremont

****

IN THIS ISSUE

****

1985 HUG OFFICERS

President ---	MARK CRUMP .... unlisted	Secretary -	JIM HUMPHREY . 465-6525
VP/Membership	DON LEWIS 353-5295	Treasurer -	JERRY ILLING . 664-7059
VP/Program --	DAVID SHOLMIRE 482-0186	Librarian -	LARRY PIPKIN . 499-9991
VP/S.I.G. ---	ROGERS MILLS .. 930-0810	TIBBS/SysOp	BILL KNECHT .. 473-5713
Exec. Asst. -	TOM JAY 850-0222	Editor ----	PHIL POXON .. 973-2362

****

THIS NEWSLETTER IS PUBLISHED MONTHLY BY THE HOUSTON USERS' GROUP. ANY OPINIONS OR ENDORSEMENTS ARE THOSE OF THE AUTHOR, AND MAY NOT NECESSARILY REFLECT THE OFFICIAL OPINION OF 'HUG'. PERMISSION TO REPRINT GRANTED TO OTHER USER GROUPS. SUBSCRIPTION IS FREE WITH MEMBERSHIP.

CODES				
binary			Char	Notes
bit no.	dec.	hex.		
76543210				
00000000	0	>00		
00000001	1	>01		
00000010	2	>02		
00000011	3	>03		
00000100	4	>04		
00000101	5	>05		
00000110	6	>06		2*
00000111	7	>07		
00001000	8	>08		
00001001	9	>09		
00001010	10	>0A		
00001011	11	>0B		
00001100	12	>0C		
00001101	13	>0D		
00001110	14	>0E		
00001111	15	>0F		
00010000	16	>10		
00010001	17	>11		
00010010	18	>12		
00010011	19	>13		
00010100	20	>14		
00010101	21	>15		
00010110	22	>16		
00010111	23	>17		
00011000	24	>18		
00011001	25	>19		
00011010	26	>1A		
00011011	27	>1B		
00011100	28	>1C		
00011101	29	>1D		
00011110	30	>1E		A*
00011111	31	>1F		B*
00100000	32	>20		C*
00100001	33	>21	!	
00100010	34	>22	"	
00100011	35	>23	#	
00100100	36	>24	$	
00100101	37	>25	%	
00100110	38	>26	&	
00100111	39	>27	'	
00101000	40	>28	(	
00101001	41	>29	)	
00101010	42	>2A	*	
00101011	43	>2B	+	
00101100	44	>2C	,	
00101101	45	>2D	-	
00101110	46	>2E	.	
00101111	47	>2F	/	

CODES				
binary			Char	Notes
bit no.	dec.	hex.		
76543210				
00110000	48	>30	0	
00110001	49	>31	1	
00110010	50	>32	2	
00110011	51	>33	3	
00110100	52	>34	4	
00110101	53	>35	5	4*
00110110	54	>36	6	
00110111	55	>37	7	
00111000	56	>38	8	
00111001	57	>39	9	
00111010	58	>3A	:	
00111011	59	>3B	;	
00111100	60	>3C	<	
00111101	61	>3D	=	
00111110	62	>3E	>	
00111111	63	>3F	?	
01000000	64	>40	@	
01000001	65	>41	A	
01000010	66	>42	B	
01000011	67	>43	C	
01000100	68	>44	D	
01000101	69	>45	E	5*
01000110	70	>46	F	
01000111	71	>47	G	
01001000	72	>48	H	
01001001	73	>49	I	
01001010	74	>4A	J	
01001011	75	>4B	K	
01001100	76	>4C	L	
01001101	77	>4D	M	
01001110	78	>4E	N	
01001111	79	>4F	O	
01010000	80	>50	P	
01010001	81	>51	Q	
01010010	82	>52	R	
01010011	83	>53	S	
01010100	84	>54	T	
01010101	85	>55	U	
01010110	86	>56	V	
01010111	87	>57	W	
01011000	88	>58	X	
01011001	89	>59	Y	
01011010	90	>5A	Z	
01011011	91	>5B	[	6*
01011100	92	>5C	\	
01011101	93	>5D	]	
01011110	94	>5E	^	
01011111	95	>5F	_	

CODES				
binary			Char	Notes
bit no.	dec.	hex.		
76543210				
01100000	96	>60	'	
01100001	97	>61	a	
01100010	98	>62	b	
01100011	99	>63	c	
01100100	100	>64	d	
01100101	101	>65	e	7*
01100110	102	>66	f	
01100111	103	>67	g	
01101000	104	>68	h	
01101001	105	>69	i	
01101010	106	>6A	j	
01101011	107	>6B	k	
01101100	108	>6C	l	
01101101	109	>6D	m	
01101110	110	>6E	n	
01101111	111	>6F	o	
01110000	112	>70	p	
01110001	113	>71	q	
01110010	114	>72	r	
01110011	115	>73	s	
01110100	116	>74	t	
01110101	117	>75	u	
01110110	118	>76	v	
01110111	119	>77	w	
01111000	120	>78	x	
01111001	121	>79	y	
01111010	122	>7A	z	
01111011	123	>7B	{	
01111100	124	>7C	|	
01111101	125	>7D	}	
01111110	126	>7E	~	
01111111	127	>7F		8*

A C C I I C O D E S

Ben Takach,

Composed by: TI-WRITER

File Name: ASCII

Sydney News Digest

Newsletter of

TI Sydney Users' Group

December 1985

THIS IS A MEMORY MAP OF SCRATCH PAD RAM

(These addresses are not decoded in the console)
(All addresses beginning with >80 through >83 are the same)

PEEK ADDRESS	MEMORY LOC	CONTENTS
-32000	>8300	XB TEMPORARY STORAGE AREA
-32000	>8300	temporary variable
-31998	>8302	temporary variable
-31996	>8304	temporary variable
-31994	>8306	temporary variable-record length on file access
-31992	>8308	temporary variable-address of sprite attribute list
-31990	>830A	temporary variable
-31988	>830C	temporary variable
-31986	>830E	temporary variable-increment value for Auto Num
-31984	>8310	temporary variable-used in CALL LINK parameter passing
-31982	>8312	temporary variable-used by CHAR type statements
-31980	>8314	temporary variable-copy of VDP reg 1 for some commands
-31978	>8316	temporary variable-DSR Link flag for some commands
-31976	>8318	XB PERMINENT STORAGE AREA
-31976	>8318	Used by LINK, LOAD & rtn control to Basic-also Str space bgn
-31974	>831A	Points to 1st free add in VDP RAM -also Str space end
-31972	>831C	Points to allocated str space -PAB ERROR-Temp string pointer
-31970	>831E	Start of current statement
-31968	>8320	Current screen address
-31966	>8322	Return error code from Assembly Language Code
-31964	>8324	VDP value stack base pointer
-31962	>8326	Return address from Assembly Language Code
-31960	>8328	NUD Table for Assembly Language Code
-31958	>832A	Ending screen display pointer
-31956	>832C	Program text or token code pointer
-31954	>832E	Pointer to current line number in line number table
-31952	>8330	Start of line number table pointer
-31950	>8332	End of Line number table pointer
-31948	>8334	Data pointer for read
-31946	>8336	Line number table pointer for read
-31944	>8338	Address of intrinsic Poly constants
-31942	>833A	Subprogram symbol table pointer
-31940	>833C	PAB address in VDP RAM (first link) PAB list
-31938	>833E	Symbol table pointer
-31936	>8340	VDP Ram free space pointer
-31934	>8342	Current char/token
-31932	>8344	Extended Basic Program RUN=255 STOP=0 (w/o 'ready')
-31931	>8345	Extended Basic System Flags Bit 0 1=Auto-num Bit 4 1=Edit mode 1 1=On Break next 5 1=On Warning Stop 2 6 1=On Warning Next 3 1=Trace 7
-31930	>8346	Crunch buffer destruction level
-31928	>8348	Last subprogram block on stack
-31926	>834A	FLOATING POINT and DSR usage, 36 bytes
-31926	>834A	FAC (floating point accumulator PAB I/O OPCODE
-31925	>834B	for floating point routines PAB FLAG/STATUS
-31924	>834C	this area holds a number in PAB DATA BUFFER ADDRESS
-31922	>834E	radix 100 notation PAB LOGICAL REC LENGTH
-31921	>834F	PAB CHARACTER COUNT
-31920	>8350	PAB RECORD NUMBER
-31918	>8352	PAB SCREEN OFFSET
-31917	>8353	PAB OPTION LENGTH
-31916	>8354	floating point error code PAB DEVICE LENGTH
-31914	>8356	Subroutine pointer/DSR's pnts to 1st char after PAB in VDP
-31912	>8358	DSR
-31910	>835A	DSR
-31908	>835C	ARG (Floating point argument) DSR
-31907	>835E	and DSR usage DSR
		DSR
-31892	>836C	FPERAD (float pnt err add in GROM?) DSR
-31891	>836D	set to >08 for DSR call DSR

```
-31890   >836E     Interpreter and Floating Point GPL Value Stack pointer
-31888   >8370     Highest Available Address in VDP RAM
-31886   >8372     LSByte of Data Stack Pointer            A0=(>83A0)
-31885   >8373     LSByte of Subroutine Stack Pointer      80=(>8380)
-31884   >8374     Keyboard Number to be scanned    Default=0
-31883   >8375     ASCII code detected by SCAN routine-also SGN for float/point
-31882   >8376     Joystick Y-Status by SCAN routine  -also EXP for float/point
-31881   >8377     Joystick X-Status by SCAN routine
-31880   >8378     Random Number Generator                 RND's >0->63 (0-99)
-31879   >8379     VDP interrupt timer                           >0->FF (0-255)
-31878   >837A     Highest Sprite # in auto-motion               >0->20 (0-32)
-31877   >837B     Copy of VDP Status register
-31876   >837C     GPL Status byte (Set to 0 for a DSR CALL) (>20=key press)
-31875   >837D     Character Buffer Byte to VDP RAM screen table
-31874   >837E     Points to the current Row on the screen
-31873   >837F     Points to the current Column on the screen
-------------------------------------------------------------------------------
-31872 >8380      THE DEFAULT SUBROUTINE STACK (Used by GPL Routines)
-31872   >8380     Reserved for Basics interpreter
-31870   >8382     Reserved for Basics interpreter
-31868   >8384     Reserved Highest Address in Expansion Memory
-31866   >8386     Reserved Highest Free Address in Mem-Expansion
-31864   >8388     Reserved for the Basics Interpreter  Sub stack base
-31863   >8389     Reserved for the Basics Interpreter  Exp-Memory flag
-31862   >838A     Return Address Stack for GROM Subroutines
                   (current GROM Address pushed to top of stack during Key Scan
-31842   >839E
-------------------------------------------------------------------------------
-31840 >83A0      THE DEFAULT DATA STACK (Used by GPL routines)
                    This area holds various information according to the GROM
                    routine being executed
-31810 >83BF
-------------------------------------------------------------------------------
-31808 >83C0      INTERRUPT WORKSPACE REGISTERS
-31808   >83C0      R0   Random number seed
-31806   >83C2      R1   Bit 0 1=disable All of the following
                         Bit 1 1=disable Auto Sprite Motion
                         Bit 2 1=disable Auto Sound Processing
                         Bit 3 1=disable the QUIT key
-31804   >83C4      R2   ISR Hook -Start address of User Interrupt Routine
-31802   >83C6      R3   Reserved for Keyboard state and debounce info
-31800   >83C8      R4   Reserved for Keyboard state and debounce info
-31798   >83CA      R5   Reserved for Keyboard state and debounce info
-31796   >83CC      R6   Pointer to Sound Table - also see >83FD
-31794   >83CE      R7   Number of sound bytes for Auto Sound Processing (0100)
-31792   >83D0      R8   Varies (>0000 for Cassette DSR Link)
-31790   >83D2      R9   Varies
-31788   >83D4      R10  Contents of VDP register 1 (Used for Key Scan)
-31786   >83D6      R11  Screen Time Out Counter (blanks when incremented to 0)
-31784   >83D8      R12  Return Address Saved by the Scan Routine (Old Reg 11)
-31782   >83DA      R13  Return WS for context switch (RTWP)
-31780   >83DC      R14  Return PC for context switch (RTWP)
-31778   >83DE      R15  Return ST for context switch (RTWP)
-------------------------------------------------------------------------------
-31776 >83E0      GPL WORKSPACE REGISTERS  (Registers used by GPL interpreter)
-31776   >83E0      R0   Varies  Note:R0-R7,R11 and R12
-31774   >83E2      R1   Varies       are modified by Key Scan
-31772   >83E4      R2   Varies
-31770   >83E6      R3   Varies
-31768   >83E8      R4   Varies
-31766   >83EA      R5   Varies     Used by the Interrupt Routine
-31764   >83EC      R6   Varies     Used by the Interrupt Routine
-31762   >83EE      R7   Varies     Used by the Interrupt Routine
-31760   >83F0      R8   Cleared on Return from Interrupt Routine
-31758   >83F2      R9   GPL Interpreter use
-31756   >83F4      R10  GPL Interpreter use
-31754   >83F6      R11  Return Address for BL instruction and User Interrupt
-31752   >83FB      R12  Varies - CRU Base Address for key scan and DSR's
-31750   >83FA      R13  GROM/GRAM read data port (9800)
-31748   >83FC      R14  Status Flags
                         Bits 0-7 Control the cursor blink speed &
                                  Auto Sound sprocessing. The value in this
                                  byte increments the counter at >8379
-31746   >83FD           Bit  0                          4  1=16K VDP RAM
                              1                          5
                              2  1=Cass interrupt Timer  6  1=M'color mode
                              3  1=Cass Verify           7  Sound Table Loc
                                                            (1=VDP 0=GROM)
-31744   >83FE      R15  VDP write address port (8C02)
-------------------------------------------------------------------------------
```

TIPS FROM THE TIGERCUB

#29

TIGERCUB SOFTWARE
156 Collingwood Ave.
Columbus, OH 43213

Over 130 original programs in Basic and Extended Basic, available on casette or disk, only $3.00 each plus $1.50 per order for PPM. Entertainment, education, programmer's utilities. Descriptive catalog $1.00, deductable from your first order.

Tips from The Tigercub, a full disk containing the complete contents of this newsletter Nos. 1 through 14, 50 original programs and files, just $15 postpaid.

Tips from the Tigercub Vol. 2, another diskfull, complete contents of Nos. 15 through 24, over 60 files and programs, also just $15 postpaid. Or, both for $27 postpaid.

Nuts & Bolts (No. 1), a full disk of 100 Extended Basic utility subprograms in merge format, ready to merge into your own programs. Plus the Tigercub Menuloader, a tutorial on using subprograms, and 5 pages of documentation with an example of the use of each subprogram. All for just $19.95 postpaid.

Nuts & Bolts No. 2, another full disk of 100 utility subprograms in merge format, all new and fully compatible with the last, and with 10 pages of documentation and examples. Also $19.95 postpaid, or both Nuts Bolts disks for $37 postpaid.

Tigercub Full Disk Collections, just $12 postpaid! Each of these contains either 5 or 6 of my regular $3 catalog programs, and the remaining disk space has been filled with some of the best public domain programs of the same category. I am NOT selling public domain programs - my own programs on these disks are greatly discounted from their usual price, and the public domain is a FREE bonus!

TIGERCUB'S BEST
PROGRAMMING TUTOR
PROGRAMMER'S UTILITIES
BRAIN GAMES
BRAIN TEASERS
BRAIN BUSTERS!
MANEUVERING GAMES
ACTION GAMES
REFLEX AND CONCENTRATION
TWO-PLAYER GAMES
KID'S GAMES
MORE GAMES
WORD GAMES
ELEMENTARY MATH
MIDDLE/HIGH SCHOOL MATH
VOCABULARY AND READING
MUSICAL EDUCATION
KALEIDOSCOPES AND DISPLAYS

For descriptions of these send a dollar for my catalog!

The offer made last month is still good until 1 January - a 10% rebate directly to the user group if one of their members mentions the user group when ordering from me. So far, I've had only 1 responses - and I suspect that 8 or 9 of those didn't even know about the offer!

I goofed again. In the I/O ERROR routine in Tips #28, the ON ERROR STOP will do no good in the place where I put it. It should be placed after the file is opened in line 100 so that it will become the current error trap if the file is opened correctly.

And the CALL KEY example in Tips #28 will look better if R=14. A couple of very knowledgeable programmers have written to tell me that I was wrong, and the manual is right, about CALL KEY status -1. They say that -1 simply means that the same key is being pressed as was pressed during the last keyscan, and that it could have been released and repressed in the interim. This may be, but try this routine and see if you can release and repress a key without getting a status code 0 (no key pressed) and status code 1 (different key pressed) before another status code -1.

```
100 CALL KEY(0,K,S):: PRINT
K,S :: GOTO 100
```

George Steffen has responded to the challenge in the last Tips, by publishing in the LA 99ers Topics a remarkably compact routine to translate the internal format string representation of numeric data back into numbers. The following lines will update the Menu Loader accordingly.

```
100 !by A. Kludge/M. Gordon/
T. Boisseau/J. Peterson/G. S
teffen/etc.Version #8, 11/85
140 @,@@,A,A$,B,C,D$,E,F,FLA
G,I,J,K,KD,KK,M,M$,N$,NN,P,P
$,PG$(),PP,PP$,Q$,S,ST,T$(),
TT,VT,V(,),W$,X,X$,Y,K2,S2
810 F=1 :: E=ASC(SEG$(M$,1,1
)):: M=ASC(SEG$(M$,2,1)):: I
F E=0 AND M=0 THEN GOTO 817
ELSE IF E>128 AND M>128 THEN
 F=-1 :: E=255-E :: M=256-M
815 FOR I=1 TO 6 :: M=M+(ASC
(SEG$(M$,I+2,1)))/100^I :: N
EXT I :: M=M*F*100^(E-64)
817 PRINT #PP:M
870 FOR P=1 TO NN-1 :: PRINT
 #2:PG$(P);TAB(15);V(P,3);TA
B(20);T$(ABS(V(P,1)));TAB(25
);V(P,2);TAB(31);CHR$(89*ABS
(V(P,1)<0)):: NEXT P :: CLOS
E #2
```

The change in the last line is my own, because it was pointed out to me that the catalog output to the printer did not indicate protected files.

That last line is a good example of the power of relational expressions to accomplish compact programming. The variable V(P,1) picks up its value from the variable A which is read from the disk directory in line 350. This is a number from 1 to 5, indicating the type of file, and if the file is write-protected the number is negative. A true expression has a relational value of -1. If the file is protected, V(P,1)<0 is true, and its value is -1, converted by ABS to +1 and multiplied by 89 to give ASCII 89, converted by CHR$ to "Y". If not protected, V(P,1) is a positive number, V(P,1)<0 is false and has a relational value of 0; 89 times 0 is still 0, and CHR$(0) prints nothing.

George also mentioned in a letter that my remarks on the UPDATE mode applied only to VARIABLE files; that RESTORE without a number,to return the record pointer to the beginning of a file, works only with VARIABLE files; that RESTORE with a number works only with RELATIVE files; and that therefore the only way to RESTORE a SEQUENTIAL FIXED file is to close it and reopen it.

On trying this out, I find that you can write to a FIXED SEQUENTIAL file and still be able to read the following records - but you can't simply "read a record, change it in some way, and then write the altered record back out on the file", as the Reference Guide indicates, because you will change the record FOLLOWING the one you read! It is possible to UPDATE a FIXED SEQUENTIAL file without reading it all into an array and writing it back out, but you must read sequentially to the record you want, close the file, reopen the file, read back to the record just before the one you want to update, then write in the updated record.

I have received several other suggestions regarding the Menu Loader, too many to describe here. You can all modify it to your own tastes and needs. Remember to turn off the pre-scan and ON ERROR while you're working on it, then add any new variable names or CALLs to the pre-scan. And remember, that last line MUST be the LAST line of the program! You can resequence it higher, and change the GOTO accordingly, but don't put anything after it!

I did change my version to slash the zero, since this will carry over into a program that is loaded. If you do this, be sure to add a CALL CHAR to the list in line 150!

```
190 CALL CLEAR :: FOR S=1 TO
 14 :: CALL COLOR(S,7,16)::
NEXT S :: CALL COLOR(0,2,16)
:: CALL CHAR(48,"003A444C546
444B8")
```

When you just want to load a program, waiting for it to be read from the disk directory can be a drag. And, you may have trouble recognizing the filename. So, here is the Tigercub Quickloader which I have placed on all my Collection Disks.

First you will need Catwriter, another program that writes a program. This

one will read the disk directory, ignore everything other than programs, ask you for a complete program name for each filename, and write all that into a MERGE format program called CATMERGE.

```
100 !CATWRITER by Jim Peters
on
110 OPEN #1:"DSK1.",INPUT ,R
ELATIVE,INTERNAL :: INPUT #1
:N$,A,J,K :: OPEN #2:"DSK1.C
ATMERGE",VARIABLE 163 :: LN=
1000 :: FN=1100
120 X=X+1 :: INPUT #1:P$,A,J
,B :: IF LEN(P$)=0 THEN 160
 :: IF ABS(A)=5 OR ABS(A)=4 A
ND B=254 THEN 130 ELSE X=X-1
 :: GOTO 120
130 DISPLAY AT(12,1)ERASE AL
L:P$;"    PROGRAM NAME?" ::
ACCEPT AT(14,1)SIZE(25):F$
140 PRINT #2:CHR$(INT(FN/256
))&CHR$(FN-256*INT(FN/256))&
CHR$(147)&CHR$(200)&CHR$(LEN
(F$))&F$&CHR$(0):: FN=FN+1
150 M$=M$&CHR$(200)&CHR$(LEN
(P$))&P$&CHR$(179):: IF X<11
 THEN 120
160 IF M$="" THEN 180
170 PRINT #2:CHR$(INT(LN/256
))&CHR$(LN-256*INT(LN/256))&
CHR$(147)&SEG$(M$,1,LEN(M$)-
1)&CHR$(0):: LN=LN+1 :: M$="
" :: X=0 :: IF LEN(P$)<>0 TH
EN 120
180 PRINT #2:CHR$(INT(LN/256
))&CHR$(LN-256*INT(LN/256))&
CHR$(147)&CHR$(200)&CHR$(3)&
"END"&CHR$(0)
190 PRINT #2:CHR$(255)&CHR$(
255):: CLOSE #1 :: CLOSE #2
```

Next, key in the Quickloader. Do not change the line numbers, do not RESequence, because CATMERGE will be merged into the middle of it and that last line must be the last. Then, enter MERGE DSK1.CATMERGE and then SAVE DSK1.LOAD .

```
100 CALL CLEAR :: DIM M$(48)
:: CALL CHAR(94,"3C4299A1A19
9423C"):: CALL SCREEN(2):: F
OR SET=1 TO 14 :: CALL COLOR
(SET,15,1):: NEXT SET :: DIS
PLAY AT(1,4):"TIGERCUB QUICK
LOADER"
110 X=X+1 :: READ M$(X):: IF
 M$(X)<>"END" THEN 110
115 CALL PEEK(8198,A):: IF A
<>17 THEN CALL INIT
120 R=3 :: FOR J=1 TO X-1 ::
 READ X$ :: DISPLAY AT(R,1):
STR$(J);TAB(4);X$ :: R=R+1 :
: IF R<23 THEN 150
130 DISPLAY AT(24,1):"CHOICE
? OR 0 TO CONTINUE 0" :: ACC
EPT AT(24,26)VALIDATE(DIGIT)
SIZE(-2):N
140 IF N<>0 THEN 155 :: R=3
150 NEXT J :: DISPLAY AT(24,
1):"CHOICE?" :: ACCEPT AT(24
,9)VALIDATE(DIGIT):N
160 IF SEG$(M$(N),LEN(M$(N))
,1)="*" THEN DISPLAY AT(12,1
)ERASE ALL:"Return to BASIC"
: :"Type OLD DSK1."&M$(N)::
STOP
170 CALL CHARSET :: CALL CLE
AR :: CALL SCREEN(8):: CALL
PEEK(-31952,A,B):: CALL PEEK
(A*256+B-65534,A,B):: C=A*25
6+B-65534 :: A$="DSK1."&M$(N
):: CALL LOAD(C,LEN(A$))
180 FOR J=1 TO LEN(A$):: CAL
L LOAD(C+J,ASC(SEG$(A$,J,1))
):: NEXT J :: CALL LOAD(C+J,
0):: GOTO 300000
300000 RUN "DSK1.1234567890"
```

If you don't want to give your Basic-only programs a filename ending in an asterisk, you can leave out that warning routine, or you can modify it to warn of E/A or MiniMemory programs. If Catwriter has picked up any unloadable program-format files, etc., just delete them from the DATA lines.

The first issue of the GENIAL TRAVeleR has arrived, and it is SUPERB! This is a magazine-on-a-disk, a SS/SD flippy loaded with 700 sectors of some of the finest articles and programs you'll ever see! And the programs are ready to run, you don't have to key anything in. The subscription price, until the end of 1985 at least, is $30 for 6 issues, which computes out to $5 per disk - many of you are paying your own user group that much for a one-sided disk of public domain!

If the subscribers will only have the guts to refuse to let their friends copy this for free, this venture will surely survive and contribute greatly to the advancement of the TI. The address is -

GENIAL COMPUTERWARE, 835 Green Valley Drive, Philadelphia PA 19128.

Gene Burchfield asked if I had a program to print banners vertically. I had never heard of such a thing, so I wrote one.

```
100 DISPLAY AT(12,1)ERASE AL
L:"TIGERCUB STREAMER PRINTER
" !by Jim Peterson
110 DATA 0000,0001,0010,0011
,0100,0101,0110,0111,1000,10
01,1010,1011,1100,1101,1110,
1111
120 RESTORE 110 :: DIM B$(16
):: FOR J=1 TO 16 :: READ B$
(J):: NEXT J :: P$(0)=" " ::
 P$(1)=CHR$(230)
130 INPUT "TEXT TO BE PRINTE
D? ":T$ :: PRINT :: INPUT "P
RINTER DESIGNATION? ":PD$ ::
 OPEN #1:PD$
140 PRINT :: INPUT "SIZE? (1
-10) ":Z :: IF Z<1 OR Z>10 T
HEN 140
150 FOR J=1 TO LEN(T$):: A=A
SC(SEG$(T$,J,1)):: IF A=32 T
HEN GOTO 200
160 CALL CHARPAT(A,H$):: FOR
 N=1 TO 15 STEP 2 :: K$=SEG$
(H$,N,2):: FOR L=1 TO 2 :: L
$=SEG$(K$,L,1):: B=POS("0123
456789ABCDEF",L$,1)
170 M$=B$(B):: FOR M=1 TO 4
:: N=VAL(SEG$(M$,M,1)):: N$=
N$&RPT$(P$(N),Z):: NEXT M
180 NEXT L :: FOR Q=1 TO Z/2
+.5 :: PRINT #1:TAB((81-Z*8)
/2+.5);N$ :: NEXT Q :: N$=""
 :: NEXT W :: FOR R=1 TO Z/2
+.5 :: PRINT #1:"" :: NEXT R
190 NEXT J :: STOP
200 FOR T=1 TO Z*4 :: PRINT
#1:"" :: NEXT T :: GOTO 190
210 CALL KEY(0,K,S):: IF S=0
 THEN 210 ELSE RETURN
```

If your printer doesn't have the special characters of the Gemini, substitute 88 instead of 230 in line 120, to print X's, or whatever else you want. If you do have the special characters, try some others, such as 239, for this and other graphics printing programs. This routine will print a handy reference chart of them.

```
100 IMAGE ### #   ### #   ##
# #   ### #   ### #   ### #
110 P$=RPT$(CHR$(251)&CHR$(2
53),21):: X=0
120 OPEN #1:"PIO" :: PRINT #
1:CHR$(27);"E"
130 PRINT #1:P$:"  ASCII COD
ES FOR GEMINI SPECIAL CHARAC
TERS":P$
140 FOR J=160 TO 175 :: K=J-
X
150 PRINT #1,USING 100:K,CHR
$(J),K+16,CHR$(J+16),K+32,CH
R$(J+32),K+48,CHR$(J+48),K+6
4,CHR$(J+64),K+80,CHR$(J+80)
:: NEXT J
160 IF FLAG=1 THEN STOP ELSE
 FLAG=1 :: PRINT #1:"":"":P$
:"TI-WRITER CODES FOR GEMINI
 SPECIAL CHARACTERS":P$ :: X
=128 :: GOTO 140
```

Another one that just looks pretty -

```
100 !KALEIDOSPRITES by Jim P
eterson
110 CALL CLEAR :: FOR CH=100
 TO 128 STEP 4 :: FOR L=1 TO
 4 :: RANDOMIZE :: X$=SEG$("
0018243C425A667E8199A5BDC3DB
E7FF",INT(16*RND+1)*2-1,2)
120 B$=B$&X$ :: C$=X$&C$ ::
NEXT L :: CALL CHAR(CH,RPT$(
B$&C$,4)):: B$,C$="" :: NEXT
 CH :: Z=2 :: CALL SCREEN(5)
130 CALL MAGNIFY(Z):: K=1 ::
 FOR J=1 TO 7 :: S=96+4*J ::
 R=16*J :: C=100*RND+20
140 IF J>5 AND Z=4 THEN T=5
:: GOTO 160
150 T=INT(15*RND+2):: IF T=5
 THEN 150
160 CALL SPRITE(#K,S,T,R,C,#
K+1,S,T,177-R,C,#K+2,S,T,R,2
41-C,#K+3,S,T,177-R,241-C)::
 K=K+4 :: NEXT J
170 Z=INT(2*RND+1)*2 :: GOTO
 130
```

```
100 !DISK MATCHER by Jim Pet
erson
110 DISPLAY AT(8,9)ERASE ALL
:"DISK MATCHER": : : :" To c
ompare a backup disk":"with
a master and list any":"file
s found on one but not"
120 DISPLAY AT(15,1):"on the
 other.": : : :"       Press
 any key"
130 CALL KEY(0,K,S):: IF S=0
 THEN 130
140 DISPLAY AT(12,1)ERASE AL
L:"INSERT MASTER - PRESS ENT
ER" :: CALL KEY(0,K,S):: IF
S=0 THEN 140
150 OPEN #1:"DSK1.",INPUT ,R
ELATIVE,INTERNAL :: INPUT #1
:D1$,A,J,K :: DIM F1$(127)
160 X=X+1 :: INPUT #1:F1$(X)
,A,J,B :: IF LEN(F1$(X))<>0
THEN 160 ELSE CLOSE #1
170 DISPLAY AT(12,1)ERASE AL
L:"INSERT BACKUP DISK": :"PR
ESS ENTER" :: CALL KEY(0,K,S
):: IF S=0 THEN 170
180 OPEN #1:"DSK1.",INPUT ,R
ELATIVE,INTERNAL :: INPUT #1
:D2$,A,J,K :: DIM F2$(127)
190 Y=Y+1 :: INPUT #1:F2$(Y)
,A,J,B :: IF LEN(F2$(Y))<>0
THEN 190 ELSE CLOSE #1
200 DIM F(127):: FOR J=1 TO
X :: FOR L=1 TO Y :: IF F2$(
L)=F1$(J)THEN F(L)=1 :: GOTO
 220
210 NEXT L :: PRINT F1$(J);"
 NOT ON BACKUP"
220 NEXT J
230 FOR M=1 TO Y :: IF F(M)=
0 THEN PRINT F2$(M);" NOT ON
 MASTER"
240 NEXT M :: END
```

A very useful tip from Jim Swedlow, in the Orange County ROM newsletter - INPUT respects any trailing print separator on a preceding PRINT command. Try it -

```
100 PRINT TAB(20);:: INPUT B
$
```

MEMORY FULL IN LINE 480
Jim Peterson

THE ORPHAN CHRONICLES

THE BEHIND THE SCENES STORY THAT SHOCKED OVER 2 MILLION PEOPLE.

RONALD G. ALBRIGHT JR. M.D. TAKES US THROUGH THE PAST PRESENT AND FUTURE OF THE TI 99/4A COMMUNITY. WITH HIS MANY INTERVIEWS AND OBSERVATIONS, TOUCHED WITH WITTY HUMOR, THE HISTORY UNFOLDS.

THE APPENDICES PROVIDE AN INVALUABLE REFERENCE WITH LISTS OF:

TI PRODUCED SOFTWARE, 129 TI BULLETIN BOARD NUMBERS, HARDWARE PRODUCERS, SOFTWARE PRODUCERS, MAIL ORDER DISTRIBUTORS, PUBLICATIONS, FREEWARE PRODUCERS, BOOKS AND USERS GROUPS FROM AROUND THE WORLD.

A Few Excerpts from THE ORPHAN CHRONICLES

..... There are numerous stories surrounding Kaplan's relationships with his writers, recruited from grassroot TI users. Most were nonprofessional writers that Kaplan recalls "giving immeasurable professional help" to improve their writing skills. One of the most interesting sidelights of the 99'er Magazine tale was the story of "Regena". The byline "Regena" first appeared in the premier issue of 99'er Magazine and, in succeeding issues, became associated with some of the best programs that appeared in the magazine. Then, according to Cheryl Whitelaw, the real "Regena" (Regena was Whitelaw's middle name; she used the name initially to give her an easy-to-remember byline), Kaplan had the idea to make the identity of Regena a mystery. He played it up with frequent "Who is Regena?" fillers in the magazine. According to Whitelaw, the idea was acceptable to begin with, but, then, as it was drawn out longer and longer, it became a burden. Kaplan would not release the address of Regena to anyone and would not forward mail to her. Kaplan told her, finally, that her identity would be made known at a gala "unveiling" at the 99'er Magazine-sponsored TI-Fest in October, 1982. But, as events would have it, not only was Regena not unveiled there, she was not even invited to attend. Further, Kaplan backed out of a deal with Whitelaw to produce her typing tutor software as promised.

..... As the recollections of other "former" 99'er Magazine authors has confirmed, their former editor was not very skilled at either employee or subscriber relations. Home Computer Magazine has not published the identity of Regena to this day.

..... [Dave Wakeley, past president of the Chicago's User Group, describes how the first TI Faire came to pass] "Having lots of cash handy and virtually no bills, we decided to hold a 'TI Faire' where we would invite vendors to come and set up booths and sell TI hardware and software, and to simultaneously hold various classes on programming the machine. We combed the pages of 99er for the addresses of vendors and got commitments from 14. The only other such gathering of which we were aware had been 99er's show in San Francisco the previous year. We did not want to try and compete with them, so Sam [Pincus] called Gary Kaplan and he told us they would not be doing another show, but agreed to come out and be our guest speaker, to talk about the bright future of the 99/4A and the rumors of a new, powerful machine from Lubbock.

At our October meeting that year, with the TI Faire plans finalized, we hosted Ed Weist, the Texas Instruments User Group Coordinator, and his traveling software show. He wowed us with the Forti music board and TI Forth, and assured us that all was well. Then came October 28th. [Wakeley agrees with the uncanny ability of TI owners to remember "where they were when.."; Wakeley heard the announcement on the car radio while on the way to his wedding reception.] All heck broke loose in Chicago, just like everywhere else TI owners congregate. For about 3 days it was impossible to get onto our board due to TI owners seeking info about the pullout and what it meant. Despite the news, something funny happened. By coincidence, our Faire was held just two weeks after 'the announcement'. We did a little local publicity, but were totally unprepared for the nearly 1,000 people who showed up. Some vendors ran out of software in one hour. We also signed up 45 new members to the group and everyone wanted information. Almost everyone. Gary Kaplan did not show up. No explanation. A few weeks later he called Sam Pincus in a panic, asking him to write an article on the Apple. The rest, as they say, is publishing history (or is it infamy?)."

..... Another periodical that started up after the pullout of Texas Instruments was "Super 99 Monthly". Published by Bytemaster Services of Sulphur, Louisiana (near Lake Charles) and edited by Richard Mitchell, owner of Bytemaster. The 32 year old Mitchell, an accountant by training and presently working for a large law firm as systems operator of a mini-computer, bought his first 99/4A in mid-1983. He chose TI "for the 16-bit processor and the price." As he started working with the machine, he became "hooked", working and learning on his machine about 100 hours per month. Looking over the information published for the computer at that time, he was not impressed, and, when TI dropped production, the idea of a periodical first came to him. He recalls "I figured some things are best when everyone wants out, leaving a void." He was even more determined to begin a new publication when he witnessed Home Computer Magazine's disappearing act in late 1983. He planned to publish in the spring of 1984, but a large advertisement paid for and scheduled to appear in the remaining TI periodical, IUG's "Enthusiast" never happened. The money was lost when IUG went bankrupt. Not a very encouraging start for Mitchell, but he persisted.

HUG LIBRARY CATALOG ADDENDUM
January 1986

0180 NINE MEN'S MORRISXB**

A fascinating game from David Mather. The object of the game is to get 3 men lined up in a row and capture all of your opponent's men. 37 sectors

0181 ADVENTUREXB**

A fascinating adventure game by Chris & Andrew Kirmse. You must give the computer directions to guide you around and solve the mystery. Well done. 89 sectors

4137 MORISON DISKXB**

Assembly subroutines written by John Clulow. This disk really shows you the capability of your system. Requires 1 Double sided or 1 Double density disk to run. 693 sectors.

4138 CATALOGXB** Printer optional

A Public domain program written by Oliver Hebert. This program will catalog either to screen or to your printer. 14 sectors

4139 QUICK-CATALOGXB**

Quick cataloging of any disks up to DSDD. Another fine program by Oliver Hebert. Displays on screen only. 7 sectors

4140 PROGRAMMER'S AIDXB**

A handy aid to find variable line references, goto and gosub routines and unique variable references within your program. A great programming tool. Written by J.R. Dew with revisions by Paul Hutnacher, Don Lewis and Bob Floyd. 20 sectors

4141 READ DISPLAY FILESXB** Printer optional

This program by Oliver Hebert enables you to read and display DV/80 AND DF/80 files to your screen or printer. 14 sectors

4142 LABELS & ENVELOPESXB** Printer required

This program written by Cy Leonard prints labels and addresses envelopes for you. 25 sectors

4143 MASS-TRANSFERE/A required**

Another mass transferring program by Stu Olson. This program has many features and can be used with auto-dial modems. Program loads in EA/5. 73 sectors

4144 MULTIPLAN TUTORIALSDV/80** E/A or TI-Writer/Printer rqd.

Three tutorial files on how to use Microsoft Multiplan. 157 sectors

4145 SPEECH TUTORIALDV/80** E/A or TI-Writer/Printer rqd.

A great tutorial by Ron Albright on using speech on your TI 99/4A. 46 sectors

5227 EDELWEISSXB**

Beautiful music from Helen Girouard. Excellent rendition of Rodgers & Hammerstein's popular song. 16 sectors

5228 MOZARTXB**

Beautiful music by Helen Girouard. Plays "Sonata In C Minor" and "Eine Kleine Nachtmusik" by Wolfgang Amadeus Mozart. 51 sectors

PHILIP POXON
EDITOR
9122 HAMMERLY
HOUSTON TX 77080