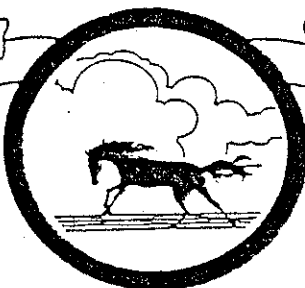


Heart of the

Bluegrass

BYTEMONGER



LEXINGTON, KENTUCKY

JUNE 1988

MEETING NOTES

by WESLEY R. RICHARDSON

The June meeting of the BLUEGRASS 99 COMPUTER SOCIETY, INC. will be held on Thursday, June 2, 1988, beginning at 7:00 P.M. at the KENTUCKY UTILITIES OPERATIONS CENTER, 500 Stone Road, Lexington, Kentucky. The library will open at 6:30 P.M. and will be open again during the break and after the meeting.

BIGBUCKS

Courson Zarger will demonstrate his BIGBUCKS program which he has written. BIGBUCKS uses a moving average for tracking stock market prices and trends. The listings for the BIGBUCKS group of programs is in three parts and will be found in the May, June and July issues of the BYTEMONGER.

FUNNELWEB V4.0

Following Courson Zarger, Wes Richardson will demonstrate Funnelweb version 4.0. Funnelweb is a versatile loader for TI-Writer, Editor Assembler, Disk Manager 1000, and Disko. It also can be configured to load other programs from a menu selection.

QUESTIONS?

The June meeting is your chance to bring any questions which you may have regarding the TI-99/4A. Please bring your questions in writing, and include a program or file on disk, if relevant. A panel of experts (anyone I can find who knows) will respond to the questions at the July 7, 1988 meeting of the Bluegrass 99 Computer Society.



TINET SOFTWARE SYSTEM

Texas Instrument International Users Network (TINET), a service of Delphi, has several interesting offers of late. You can now join TINET for \$20 off, or \$29.95. The package includes a copy of Simon & Schuster's book *Delphi: The Official Guide*, a commend card, and one hour credit for use.

Fairware authors take note! The surcharge software allows members of TINET to upload as many fairware programs as desired. You (the author) decide the price and a check is sent directly to you in 45 days by Delphi. A 10% service charge is assessed by Delphi. Dial 1-800-365-4636 or 1-617-2981. At Password, type TINET and press return.

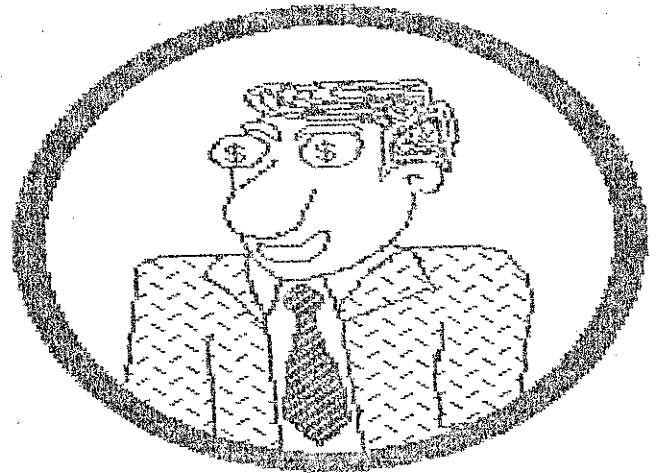
The First Annual TINET Programming Contest is in the works! Open to all users of the TI 99/4A and Myarc 9640, you may submit original work in any language available on those computers. Winners may be offered commercial software contracts by Disk Only, Asgard, Genial, etc. For rules & info write TINET Programming Contest, P.O. Box 244, Lorton, VA 22079, or mail TINET on Delphi.

B-I-I-G BUCKS

SECOND OF A SERIES

by C. ZARGER

BLUEGRASS 99 COMPUTER SOCIETY, INC.



This is the second in a series of three groups of programs which use a moving average to determine when a recent change in stock market price trends is real. IT DOES NOT PREDICT ANYTHING!!!! It merely confirms that a real change has occurred.

The programs which are required for BIGBUCKS include the following: BIGBUCKS BUCKOPEN BUCKUPDATE BUCKCHANGE BUCKPRINT and BUCKSEE. These programs will be included in the BLUEGRASS 99 COMPUTER SOCIETY newsletter the BYTEMONGER in three installments over May, June and July, 1988.

```

* 10 ! $$$$$$$$$$$$$$$$$$$$$$$$$$$$$
20 ! $ BUCKUPDATE $
30 ! $ By C Zarger $
40 ! $Bluegrass 99 Computer$
50 ! $ Society, INC $
60 ! $$$$$$$$$$$$$$$$$$$$$$$$$$$$$
100 DISPLAY AT(2,7)ERASE ALL:"B-I-I-G
Buck$": :TAB(8);"ADD A RECORD": :
"Choose one of the following:"
110 DISPLAY AT(8,4):"1 Tracking entry:
": : " 2 Buy or sell holdings:" :
: " 3 Switch holdings:" : " 4 R
ecord a dividend:"
120 DISPLAY AT(16,4):"5 Change smoothi
ng days:" : " 6 Compute interest
:" : " 7 Return to main menu": :
" Your choice? 1"
130 ACCEPT AT(22,18)SIZE(-1)VALIDATE("
1234567"):Q :: A$=RPT$(" ",56):: F
OR T=6 TO 22 STEP 2 :: DISPLAY AT(
T,1):A$ :: NEXT T
140 DISPLAY AT(9,1):"Fund (FILE) name:
": : "Data drive #: 2": : "Data date
(YMMDD):": : "Share value $"
150 DISPLAY AT(17,1):"Transaction tota
l $": : : : " Correct? (Y/N) Y"
:: IF Q=1 THEN CALL TRAK :: GOTO
100
160 IF Q=2 THEN CALL BYSEL :: GOTO 100
ELSE IF Q=3 THEN CALL SWCH :: GO
TO 100 ELSE IF Q=4 THEN CALL DVND
:: GOTO 100
170 IF Q=5 THEN CALL SMOOTH :: GOTO 10
0 ELSE IF Q=6 THEN CALL INTRST ::
GOTO 100
180 DISPLAY AT(12,1)BEEP ERASE ALL:"Fi

```

```

le update completed.": "Returning t
o MAIN MENU." :: FOR T=1 TO 500 ::
NEXT T :: RUN "DSK1.BIGBUCKS"
190 SUB TRAK
200 DISPLAY AT(4,8):"Add A Record":TAB
(7);"TRACKING ENTRY": : "Enter requ
ested data . . .": : "Fund (FILE) n
ame": : : "Data drive #: 2"
210 ACCEPT AT(9,18):N$ :: ACCEPT AT(11
,15)SIZE(-1)VALIDATE("1234"):DR ::
FN$="DSK"&STR$(DR)&". "&N$ :: SW=0
220 ACCEPT AT(13,19):DD$ :: ACCEPT AT(
15,14):CV :: ACCEPT AT(22,19)SIZE(
-1)VALIDATE("YNyn"):Q$ :: IF Q$<>"
Y" AND Q$<>"y" THEN 210
230 IF SW=0 THEN DISPLAY AT(22,1)BEEP:
"Put data disk in drive";DR:"then
press ENTER." :: ACCEPT AT(23,19):
Q$
240 OPEN #1:FN$,UPDATE,RELATIVE,INTERN
AL :: INPUT #1,REC 0:DD$,V,N,T$,I,S
,RN :: Q=RN+1 :: IF T$="E" OR T$="
e" THEN 260
250 DISPLAY AT(9,1)BEEP:"MONEY MARKET
- USE OPTION 6." :: FOR T=1 TO 500
:: NEXT T :: GOTO 380
260 IF RN<1 THEN CP=INT(N*CV*100+.5)/1
00 :: CT=V :: PRINT #1,REC Q:DD$,C
V,CT,0,CP,1 :: GOTO 340 ELSE 270
270 FOR T=0 TO -1 STEP -1 :: INPUT #1,
REC RN:ID$,IV,IT,IN,IP,TYPE :: T=(
ID$<=DD$): : IF T<0 THEN 290
280 PRINT #1,REC RN+1:ID$,IV,IT,IN,IP,
TYPE :: RN=RN-1 :: NEXT T
290 CP=INT((IP/IV)*CV*100+.5)/100 :: C
T=INT((IT+(CV-IT)*S)*10000+.5)/100
00 :: RN=RN+1 :: PRINT #1,REC RN:D
D$,CV,CT,0,CP,1
300 IF RN=Q THEN 340 ELSE RN=RN+1
310 FOR T=RN TO Q :: INPUT #1,REC T:ID
$,IV,IT,IN,IP,TYPE :: IF TYPE<>1 T
HEN 330
320 IT=INT((CT+(IV-CT)*S)*10000+.5)/10

```

PLUS!

USING TI WRITER INCLUDE FILE AND
TRANSLITERATE COMMANDS
Part One of Two Parts

By Mark Armstrong
BLUEGRASS 99 COMPUTER SOCIETY, INC.

Introduction

PLUS! is a fairware disk which will be demonstrated at the July meeting. This discussion is presented in anticipation of that demonstration. The purpose of this discussion is to briefly review the Include File (.IF) and Transliteration (.TL) commands and the Special Character Mode of TI Writer. TI Writer, hereinafter referred to as TIW, is used generically in this discussion and includes the original TI Writer and its progeny, BA Writer, TK Writer, and FW Writer.

I. Include File Command

After text is created in the Text Editor, it may be printed through the .PF function of the Text Editor or through the Formatter. Certain TIW commands are effective only when the document is printed through the Formatter; e.g., margin commands such as .LM and .RM. These Formatter commands include the Include File (.IF) command. The TIW manual presents the .IF command as being primarily for the purpose of enabling the user to print documents which exceed the size of the print buffer. The procedure for using the .IF command is to create a calling file which will call text files to be printed through the formatter. Program control is passed from the calling file to the text file. Control returns to the calling file after the text file has been printed.

For example, each chapter of a book could be a separate text file the size of each in excess of the print buffer. To print all the chapters, it is only necessary to create a calling file:

```
.IF DSKn.CHAPTER1
.IF DSKn.CHAPTER2
.IF DSKn.CHAPTER3
.IF DSKn.....
.IF DSKn.CHAPTERN
```

Even though several files are to be printed, the Formatter treats the calling file and the text files as one file. Formatter commands may be included in the calling file or in each text file. If the formatter commands are placed in the calling file, these commands will control the printing of the text files:

```
.LM 10;RM 75;FI;AD;IN +10
```

```
.IF DSKn.CHAPTER1
.IF DSKn.CHAPTER2
.IF DSKn.CHAPTER3
.IF DSKn.....
.IF DSKn.CHAPTERN
```

The margin, fill, adjust and indent commands contained in the calling file control the printing of the text files unless the called text file contains other formatter commands. When the called text file contains formatter commands, printer control passes from the calling file to the text file and the printer executes in accord with the formatter commands contained in the called text file being printed. After the text file is printed, printer control automatically returns to the calling file.

For purposes of this discussion, it is important only to note (1) that, as the TIW manual points out, the Formatter treats the calling file and the text files as one document and (2) that printer control returns to the calling file after the text file has been printed. Although not mentioned above, it is also of equal importance to note (3) that the calling file may contain text to be printed in addition to the .IF commands. PLUS! makes use of these three (3) facts to pass printing control to a text file which contains only printer control codes and no text. These printer control codes can be created in advance and are then available as the need arises.

The initial concept of PLUS! is to create a callable text file of printer control codes, which will be called by a calling file, which will contain text. Thus, the printing of the text in the calling file can be easily and exactly controlled through the printer codes in the called text file. This arrangement turns the .IF command on its head, i.e., the calling file becomes the text file and the text file becomes a printer control file. As we shall later see, the implementation of this the arrangement works quite well because the formatter treats both the calling file and the text file as a single document and printer control returns to the calling file after the called text file is read by the Formatter.

II. TRANSLITERATE COMMAND

The second TIW command used by PLUS! is the Transliteration (.TL) command. One method of addressing printers is through decimal numeric codes. For example, in Basic, the program line, PRINT #N: CHR\$(127)&CHR\$(15), will cause an Epson printer to print text in a Compressed Mode, i.e., 17.16 characters per inch. These printer codes are often referred to as Escape codes because the code is usually preceded by CHR\$(27), called an Escape code, which is nothing more than a signal to the printer that a printer command is being sent. Some Escape codes do not require CHR\$(27) to be sent. For example, on an Epson printer,

...BIGBUCKS

```

000 :: PRINT #1,REC T:ID$,IV,IT,IN
,IP,TYPE :: CT=IT
330 NEXT T
340 SW=1 :: PRINT #1,REC Ø:D$,V,N,T$,I
,S,Q :: CLOSE #1 :: FOR T=8 TO 10
STEP 2 :: DISPLAY AT(T,1):A$ :: NE
XT T
350 DISPLAY AT(17,1):"Transaction Tota
l $": : : : " Correct? (Y/N) Y"
:A$
360 DISPLAY AT(9,1)BEEP:"Another TRACK
entry?"(Y/N) Y": : "Same Fund? (Y/
N) Y" :: ACCEPT AT(9,28)SIZE(-1)VA
LIDATE("YNyn"):Q$
370 IF Q$<>"Y" AND Q$<>"y" THEN 380 EL
SE ACCEPT AT(11,18)SIZE(-1)VALIDAT
E("YNyn"):Q$ :: IF Q$<>"Y" AND Q$<
>"y" THEN 200 ELSE 220
380 SUBEND
390 SUB BYSEL
400 DISPLAY AT(4,8):"Add A Record":TAB
(7);"BUY/SELL ENTRY": : "Enter requ
ested data . . .": : "Fund (FILE) n
ame": : "Data drive #: 2"
410 DISPLAY AT(19,1):"Buy or sell? (B/
S) B"
420 ACCEPT AT(9,18)SIZE(10):N$ :: ACCE
PT AT(11,15)SIZE(-1)VALIDATE("1234
"):DR :: FN$="DSK"&STR$(DR)&"SN$
:: SW=Ø
430 ACCEPT AT(13,19):DD$ :: ACCEPT AT(
15,14):CV :: ACCEPT AT(17,20):CA :
: ACCEPT AT(19,20)SIZE(-1)VALIDATE
("BSbs"):BS$
440 IF BS$<>"B" AND BS$<>"b" THEN CA=-
CA
450 ACCEPT AT(22,19)SIZE(-1)VALIDATE("
YNyn"):Q$ :: CN=INT((CA/CV)*1000+.
5)/1000 :: IF Q$<>"Y" AND Q$<>"y"
THEN 420
460 IF SW=Ø THEN DISPLAY AT(22,1)BEEP:
"Put data disk in drive";DR:"then
press ENTER." :: ACCEPT AT(23,19):
Q$
470 OPEN #1:FN$,UPDATE,RELATIVE,INTERN
AL :: INPUT #1,REC Ø:D$,V,N,T$,I,S
,RN :: Q=RN+1
480 IF RN<1 THEN IT=V+INT((CV-V)*S*100
00+.5)/10000 :: IP=INT(N*V*100+.5)
/100 :: IN=N :: IV=V :: GOTO 510
490 FOR T=Ø TO -1 STEP -1 :: INPUT #1,
REC RN:ID$,IV,IT,IN,IP,TYPE :: T=(
ID$<=DD$):: IF T<Ø THEN 510
500 PRINT #1,REC RN+1:ID$,IV,IT,IN,IP,
TYPE :: RN=RN-1 :: NEXT T
510 CP=CA+INT((IP/IV)*CV*100+.5)/100 :
: RN=RN+1 :: PRINT #1,REC RN:DD$,C
V,IT,CN,CP,2 :: IF RN=Q THEN 540
520 RN=RN+1 :: FOR T=RN TO Q :: INPUT
#1,REC T:ID$,IV,IT,IN,IP,TYPE
530 IP=CA+INT((IP/IV)*100+.5)/100 :: P
RINT #1,REC T:ID$,IV,IT,IN,IP,TYPE
:: NEXT T
540 SW=1 :: PRINT #1,REC Ø:D$,V,N,T$,I
,S,Q :: CLOSE #1 :: A$=RPT$(" ",56
):: FOR T=8 TO 10 STEP 2 :: DISPLA
Y AT(T,1):A$ :: NEXT T
550 DISPLAY AT(9,1)BEEP:"Another BUY/S
ELL entry?":(Y/N) Y": "Same fund?
(Y/N) Y" :: ACCEPT AT(10,7)SIZE(-1
)VALIDATE("YNyn"):Q$
560 IF Q$<>"Y" AND Q$<>"y" THEN 580 EL
SE ACCEPT AT(11,18)SIZE(-1)VALIDAT
E("YNyn"):Q$
570 DISPLAY AT(22,1):" CORRECT? (Y/N
) Y" :: IF Q$<>"Y" AND Q$<>"y" THE
N 400 ELSE 430
580 DISPLAY AT(18,1):RPT$(" ",28):: SU
BEND
590 SUB SWTCH
600 SW=Ø :: DISPLAY AT(4,8):"Add A Rec
ord":TAB(5);"SWITCH FUNDS ENTRY":
: "Enter SWITCH FROM data . . ."
610 DISPLAY AT(9,1):"Fund (FILE) name:
": : "Data DRIVE #: 2"
620 ACCEPT AT(9,18)SIZE(10):N$ :: ACCE
PT AT(11,15)SIZE(-1)VALIDATE("1234
"):DR :: FN$="DSK"&STR$(DR)&"SN$
630 IF SW=Ø THEN ACCEPT AT(13,19):DD$
:: ACCEPT AT(17,20):CA
640 ACCEPT AT(15,14):CV :: ACCEPT AT(2
2,19)SIZE(-1)VALIDATE("YNyn"):Q$ :
: IF Q$<>"Y" AND Q$<>"y" THEN 620
650 CN=INT((CA/CV)*1000+.5)/1000 :: IF
SW=Ø THEN CN=-CN :: CA=-CA ELSE 6
70
660 DISPLAY AT(22,1)BEEP:"Put data dis
k in drive";DR:"then press ENTER."
:: ACCEPT AT(23,19):Q$
670 OPEN #1:FN$,UPDATE,RELATIVE,INTERN
AL :: IF EOF(1)THEN 770 ELSE INPUT
#1,REC Ø:D$,V,N,T$,I,S,RN :: Q=RN
+1
680 IF RN<1 THEN CP=INT((N+CN)*CV*100+
.5)/100 :: CT=V :: PRINT #1,REC Q:
DD$,CV,CT,CN,CP,3 :: GOTO 740
690 FOR T=Ø TO -1 STEP -1 :: INPUT #1,
REC RN:ID$,IV,IT,IN,IP,TYPE :: T=(
ID$<=DD$):: IF T<Ø THEN 710
700 PRINT #1,REC RN+1:ID$,IV,IT,IN,IP,
TYPE :: RN=RN-1 :: NEXT T
710 CP=CA+INT((IP/IV)*CV*100+.5)/100 :
: RN=RN+1 :: PRINT #1,REC RN:DD$,C
V,IT,CN,CP,3 :: IF RN=Q THEN 740 E

```

...BIGBUCKS

```

LSE RN=RN+1
720 FOR T=RN TO Q :: INPUT #1,REC T:ID
$,IV,IT,IN,IP,TYPE :: IP=CA+INT((I
P/IV)*CV*100+.5)/100
730 PRINT #1,REC T:ID$,IV,IT,IN,IP,TYP
E :: NEXT T
740 PRINT #1,REC 0:D$,V,N,T$,I,S,Q ::
CLOSE #1 :: IF SW=1 THEN 840
750 A$=RPT$(" ",56):: DISPLAY AT(7,1):
A$ :: DISPLAY AT(7,1)BEEP:"Enter S
WITCH TO data . . ." :: DISPLAY AT
(22,1):" Correct? (Y/N) Y":A$
760 SW=1 :: CA=ABS(CA):: GOTO 620
770 A$=RPT$(" ",56):: FOR T=6 TO 22 ST
EP 2 :: DISPLAY AT(T,1):A$ :: NEXT
T
780 DISPLAY AT(4,7):"OPEN NEW FILE": :
"Enter requested data . . ."
790 DISPLAY AT(11,1):"Equity or Moneym
kt? (E/M) E": "Smoothing length:"
: "Interest rate (%):" : : : "
Correct? (Y/N) Y"
800 ACCEPT AT(11,27)SIZE(-1)VALIDATE("
EMem"):T$ :: IF T$="E" OR T$="e" T
HEN ACCEPT AT(13,18):SD :: I=0 ::
S=INT((10/SD)*1000+.5)/1000 :: GOT
O 820
810 IF T$="M" OR T$="m" THEN S=0 :: AC
CEPT AT(15,19):I
820 ACCEPT AT(19,19)SIZE(-1)VALIDATE("
YNyn"):Q$ :: IF Q$<>"Y" AND Q$<>"y
" THEN 800
830 PRINT #1,REC 0:DD$,CV,CN,T$,I,S,0
:: CLOSE #1
840 SUBEND
850 SUB DVND
860 DISPLAY AT(4,8):"Add A Record":TAB
(7);"DIVIDEND ENTRY": "Enter requ
ested data . . ." :: SW=0
870 DISPLAY AT(9,1):"Fund (FILE) name:
": "Data drive #: 2"
880 DISPLAY AT(19,1):"Reinvested or Ca
sh (R/C) R" :: ACCEPT AT(9,18)SIZE
(10):N$ :: ACCEPT AT(11,15)SIZE(-1
)VALIDATE("1234"):DR
890 FN$="DSK"&STR$(DR)&". "SN$ :: ACCEP
T AT(13,19):DD$ :: ACCEPT AT(15,14
):CV :: ACCEPT AT(17,20):CA
900 ACCEPT AT(19,26)SIZE(-1)VALIDATE("
RCrc"):RC$ :: ACCEPT AT(22,19)SIZE
(-1)VALIDATE("YNyn"):Q$ :: IF Q$<>
"Y" AND Q$<>"y" THEN 890
910 IF SW=0 THEN DISPLAY AT(22,1)BEEP:
"Put data disk in drive";DR:"then
press ENTER." :: ACCEPT AT(23,19):
Q$
920 OPEN #1:FN$,UPDATE,RELATIVE,INTERN

```

```

AL :: INPUT #1,REC 0:D$,V,N,T$,I,S
,RN :: Q=RN+1
930 IF RN<1 THEN CN=INT((CA/V)*1000+.5
)/1000 :: IT,IV=V :: IN=N :: IP=IN
T(N*V*100+.5)/100 :: GOTO 960
940 FOR T=0 TO -1 STEP -1 :: INPUT #1,
REC RN:ID$,IV,IT,IN,IP,TYPE :: T=(
ID$<=DD$):: IF T<0 THEN CN=INT((CA
/IV)*1000+.5)/1000 :: GOTO 960
950 PRINT #1,REC RN+1:ID$,IV,IT,IN,IP,
TYPE :: RN=RN-1 :: NEXT T
960 RN=RN+1 :: IF RC$<>"R" AND RC$<>"r
" THEN CN,CA=0
970 CP=CA+INT((IP/IV)*CV*100+.5)/100 :
: PRINT #1,REC RN:DD$,CV,IT,CN,CP,
4 :: IF RN=Q THEN 990 ELSE RN=RN+1
980 FOR T=RN TO Q :: INPUT #1,REC T:ID
$,IV,IT,IN,IP,TYPE :: IP=CA+INT((I
P/IV)*CV*100+.5)/100 :: PRINT #1,R
EC T:ID$,IV,IT,IN,IP,TYPE :: NEXT
T
990 PRINT #1,REC 0:D$,V,N,T$,I,S,Q ::
CLOSE #1 :: A$=RPT$(" ",56):: FOR
T=8 TO 10 STEP 2 :: DISPLAY AT(T,1
):A$ :: NEXT T :: SW=1
1000 DISPLAY AT(7,1)BEEP:"Another DIVID
END entry?": "(Y/N) Y": "Same fund?
(Y/N) Y" :: ACCEPT AT(8,7)SIZE(-1)
VALIDATE("YNyn"):Q$
1010 IF Q$<>"Y" AND Q$<>"y" THEN 1020 E
LSE ACCEPT AT(9,18)SIZE(-1)VALIDAT
E("YNyn"):Q$ :: IF Q$<>"Y" AND Q$<>
"y" THEN 860 ELSE 890
1020 DISPLAY AT(18,1):A$ :: DISPLAY AT(
22,1):" Correct? (Y/N) Y":A$ ::
SUBEND
1030 SUB SMOOTH
1040 DISPLAY AT(4,8):"Add A Record":TAB
(3);"SMOOTHING LENGTH ENTRY": "En
ter requested data . . ." :: SW=0
1050 DISPLAY AT(9,1):"Fund (FILE) name:
": "Data drive #: 2"
1060 DISPLAY AT(13,1):"Data date(YMMDD
)": "Old smoothing length": "N
ew smoothing length": : : : "
Correct? (Y/N) Y"
1070 ACCEPT AT(9,18):N$ :: ACCEPT AT(11
,15)SIZE(-1)VALIDATE("1234"):DR ::
FN$="DSK"&STR$(DR)&". "SN$
1080 ACCEPT AT(13,19):DD$ :: DISPLAY AT
(19,1)BEEP:"Put data disk in drive
";DR:"then press ENTER." :: ACCEPT
AT(20,18):Q$
1090 OPEN #1:FN$,UPDATE,RELATIVE,INTERN
AL :: INPUT #1,REC 0:D$,V,N,T$,I,S
,Q :: SL=INT(10/S):: S$=STR$(SL)
1100 T=VAL(SEG$(S$,LEN(S$),1)):: IF T=5

```

PLUS! . . .

sending CHR\$(15) alone, i.e., PRINT #N: CHR\$(15), will activate the Compressed print mode. In this case, however, sending an Escape code, even when not required, is not only acceptable, but many informed programmers feel is preferable, coding. The matter is left to the user's discretion; however, including an Escape codes is quite helpful in decoding .TL commands.

An Escape code cannot be sent in a normal text line in TIW. Rather, printer commands are sent to the printer using the .TL function of the TIW. Every print character has been assigned an ASCII code. For example, the ASCII code for uppercase A is decimal 65. The effect of this assignment is to cause the printer to print an uppercase A when it receives ASCII code 65. The assignment of ASCII codes to characters is purely arbitrary; There is no inherent reason why ASCII code 65 was assigned to uppercase A. The function of the .TL command is to re-assign the ASCII code. For example, ASCII code 65 could be reassigned from upper case A to CHR\$(27)&CHR\$(15). After this re-assignment occurs, when the Formatter comes across an uppercase A in the text file, the Formatter sends the Escape code to the printer rather than an uppercase A. Note that the re-assignment occurs in the Formatter and not the Text Editor. Thus, in the Text Editor, an uppercase A may still be entered. When the text file is being printed through the Formatter, the re-assignment is made and ASCII code 65 is re-assigned from an uppercase A to CHR\$(27)&CHR\$(nn).

The re-assignment of the ASCII code is done through the .TL command. To reassign ASCII code 65 from upper case A to CHR\$(27)&CHR\$(15), the following text line is created:

```
.TL 65: 25,15
```

Of course, ASCII code 65 would not be transliterated into a print command because this would cause the Formatter to send a print command rather than an uppercase A. This would, for example, make an article on American Ardvarks difficult to read. For this reason, little used characters are usually transliterated; e.g., the tilde, ASCII code 126.

If the first concept of PLUS! is to use the .IF command to transfer print control to a text file, the second concept is to transliterate ASCII codes in a callable text file. Thus, when the calling file is printed through the Formatter, a printer code will be sent to the printer according to the transliteration which is made in the called text file. PLUS! requires the first line of the calling file to contain the .IF command. The Formatter then first reads the called text file containing the transliterations. After the called text file is read by the the Formatter, printer control is returned to the calling file. Then when the Formatter reads the ASCII code

in the calling file which was transliterated in the called text file, the corresponding print control code is sent to the printer. The Formatter does not print the transliterations in the called text file because the .TL commands are formatter commands which are not printed. The use of multiple print commands is not prohibited by the transliteration command. Thus, by including all the necessary ASCII codes in the .TL command, underline can be combined with compressed print.

III. Special Character Mode

As noted above, only little used characters are usually transliterated. Using the Special Character Mode of the TIW gives 32 ASCII codes which can be transliterated without sacrificing any characters.

The Special Character Mode permits access to ASCII codes 0 through 31. The characters assigned to these ASCII codes are non-printable characters and which would not otherwise be available for transliteration. For example, ASCII code 28 is a File Separator. Unlike the uppercase A, there is no File Separator character which can be entered in the Text Editor that when read by the Formatter could be transliterated into a printer code. Access to these non-printable characters can be made because characters typed in the Special Character Mode have decimal 64 subtracted from the ASCII code for codes in the range of 32 through 127. Thus, 64 is subtracted from uppercase A, ASCII code 65, to yield ASCII code 1 in the Special Character Mode. The Special Character Mode is accessed by pressing Control U. Entry into the Special Character Mode is confirmed by the transformation of the cursor from a blinking block to a blinking underline. After the Special Character Mode is accessed, any character with an ASCII code in the range of 32 through 127 may be pressed. Characters in this range are the full set characters found on a QWERTY keyboard; e.g., upper and lower case letters, numbers and symbols. The Special Character Mode is deselected by pressing Control U again.

There have been several articles discussing the use of the Special Character Mode to send print control codes. Recall that the printer command for condensed printing is CHR\$(15). This command can be sent to the printer using the Special Character Mode by pressing Control U, followed by Shift O and Control U again to deselect the Special Character Mode. 64 is subtracted from the ASCII code value of the character entered. Thus when uppercase O, Shift O (ASCII code 79), was entered, the ASCII code sent to the printer was 15 (79 less 64). Sending ASCII code 15 is the same as printing CHR\$(15) in a BASIC program. Indeed, sending CHR\$(15) is the way ASCII code 15 is sent by BASIC.

The Special Character Mode can also send an Escape Code. In the case of compressed print, the sequence of

THE PAPER LIBRARY

by Ed Powell

BLUEGRASS 99 COMPUTER SOCIETY, INC.

This month column is going to be computer generic so if you have to use one of those other brands at work the following may even be helpful there.

Saucy floppies may sound good but are not easily digested by the 99/4A. Here by way of J.C. and the ATICC Newsletter is how Jack Stoller of Verbatim Corp advises saving a disk if Jr. decides floppies and lasagna do mix.

1. Don't panic.
2. Carefully rinse off as much of the lasagna as you can using cold water.
3. Carefully slit one end of the protective P.V.C. jacket, holding the disk.
4. Put on a pair of white lint free cotton gloves and remove the actual disk. Rinse the disk several times to remove the particular matter.
5. Slit the jacket of a good disk and insert the questionable disk into the good jacket. Don't attempt to tape up the slit end.
6. Copy the newly made disk onto a regular disk and you should have recovered your data.

By way of Topics- LA 99ERS in Chick Marti's "Did you know that ...?" column, a suggestion he found in the PUN newsletter:

"When you are working on your video display terminal, BLINK, and you will have eliminated one of the causes of eye fatigue", says one Ohio ophthalmologist. "While using VDT," wrote Dr. Frank J. Wienstock in a letter to the Journal of the American Medical Association, "the user has a tendency to stare and decrease blinking to avoid missing anything on the screen; this, and not the screen itself, is what usually causes a sense of eye fatigue", he wrote. He concluded by stating that "Blinking provides eye lubrication and reduces that 'tired eye' feeling significantly".

PLUS! . . .

keystrokes would be Control U, Function R (left bracket), Shift O, Control U. The ASCII value of left bracket is 91 which less 64 equals 27. As stated above, the ASCII code 79 of uppercase O less 64 is 15. Thus, ASCII codes 27 and 15 are sent to the printer. Even when the Escape code contains an ASCII value greater than 31, it can be sent in the Special Character Mode. For example, the Escape code for Expanded print is CHR\$(27)&CHR\$(71) or ASCII codes 27 and 71. ASCII code 71 is the ASCII code for uppercase G. To send a double strike code using the Special Character Mode, press Control U, Function R, Control U, Shift G. The first Control U accesses the Special Character Mode, Function R returns the left bracket character, ASCII code 91, the second Control U deselects the Special Character Mode, and the ASCII code for uppercase G is 71. When the Formatter reads this line, it receives and sends ASCII code 27 to the printer. This code signals the printer that the next ASCII code received is a print command. The next ASCII code received is ASCII 71, uppercase G, and thus, the print command for double strike is received by the printer. The use of multiple print commands is not prohibited by using the Special Character Mode. By appropriate keystrokes, double strike can be combined with condensed.

PLUS! advances the use of the Special Character Mode by combining it with the Include File and Transliteration commands. The fourth and final concept behind PLUS! is to transliterate ASCII codes 0 through 31 into print control codes. The transliterations are placed in a text file which is called through the Include File command. Print control is returned to the calling file. When the Formatter reads a transliterated character, it sends the printer control code as defined in the previously called text file. The transliterated print codes are accessed in the calling file through the Special Character Mode. The primary advantage of PLUS! is to send multiple print commands to the printer with a minimum of keystrokes. In addition, because the transliterations and print commands are on a text file, they need only be created once. They may also be created in advance with as much complexity as desired. Accordingly, complex and customized document formats can be created. These document formats can be accessed with a minimum of keystrokes.

In next month's issue of the BYTEMONGER, the implementation of PLUS! will be discussed.

...BIGBUCKS

```

OR T=0 THEN 1120 ELSE IF T=1 OR T
=2 THEN SL=SL-T
1110 IF T=3 OR T=4 OR T=6 OR T=7 THEN S
L=SL-T+5 ELSE IF T=8 OR T=9 THEN S
L=SL+10-T
1120 DISPLAY AT(15,22):SL :: ACCEPT AT(
17,23):SL :: ACCEPT AT(22,19)SIZE(
-1)VALIDATE("YNyn"):Q$ :: SW=1
1130 IF Q$<>"Y" AND Q$<>"y" THEN CLOSE
#1 :: GOTO 1040 ELSE S=INT((10/SL)
*1000+.5)/1000
1140 PRINT #1,REC 0:D$,V,N,T$,I,S,Q ::
CLOSE #1 :: DISPLAY AT(19,1)BEEP:"
Another SMOOTHING entry?":(Y/N) N
"
1150 ACCEPT AT(20,7)SIZE(-1)VALIDATE("Y
Nyn"):Q$ :: IF Q$<>"Y" AND Q$<>"y"
THEN 1160 ELSE 1070
1160 DISPLAY AT(15,1):"Share value $":
"Transaction total $": "No. of s
hares": " " :: SUBEND
1170 SUB INTRST
1180 DISPLAY AT(4,9):"Add Record" TAB
(7);"COMPUTE INTEREST": "Enter re
quested data. . ."
1190 DISPLAY AT(9,4):"Fund (FILE) name:
": "Data drive #: 2": "Data date

```

```

(YMMMDD):": "Last date(YMMMDD):":
:"Interest rate (%):"
1200 DISPLAY AT(19,1)BEEP:"Put data dis
k in drive";DR:"then press ENTER."
: " Correct? (Y/N) Y"
1210 ACCEPT AT(9,18):N$ :: ACCEPT AT(11
,15)SIZE(-1)VALIDATE("1234"):DR ::
FN$="DSK"&STR$(DR)&"&N$ :: DISP
LAY AT(19,23):DR
1220 ACCEPT AT(13,19)SIZE(10):DD$ :: AC
CEPT AT(20,18):Q$ :: OPEN #1:FN$,U
PDATE,RELATIVE,INTERNAL :: INPUT #
1,REC 0:D$,V,N,T$,I,S,Q
1230 IF Q=0 THEN ID$=D$ :: IP=INT(V*N*1
00+.5)/100 :: IV=V ELSE INPUT #1,R
EC Q:ID$,IV,IT,IN,IP,TYPE
1240 DISPLAY AT(15,19):ID$ :: DISPLAY A
T(19,1):"Interest length (Days)":
" "
1250 ACCEPT AT(17,19):CI :: ACCEPT AT(1
9,24):T :: ACCEPT AT(22,19)SIZE(-1
)VALIDATE("YNyn"):Q$ :: IF Q$<>"Y"
AND Q$<>"y" THEN 1210
1260 IP=IP+IP*(CI/36500)*T :: IP=INT(IP
*1000+.5)/1000 :: PRINT #1,REC Q+1
:DD$,IV,0,IP,INT(IP*100)/100,6
1270 Q=Q+1 :: PRINT #1,REC 0:D$,V,N,T$,
I,S,Q :: CLOSE #1 :: SUBEND

```



BLUEGRASS 99 COMPUTER SOCIETY, INC.
 P.O. BOX 11866
 LEXINGTON, KENTUCKY 40578-1866

MIAMI CO. AREA 99-4A HCUG
 P.O. BOX 1194 PERU, IN 46970 C*06-88

 THE 1988 BLUEGRASS 99 COMPUTER SOCIETY, INC. ELECTIVE OFFICERS ARE:
 PRESIDENT --- NES RICHARDSON, Ph. 502/863-4336; VICE-PRESIDENT/EDITOR --- DORIS SETTLES, Ph. 606/269-5533;
 SECRETARY/EXCHANGE NEWSLETTER LIBRARIAN --- ED POWELL, Ph. 606/498-4595; TREASURER --- COURSON ZARGER, Ph. 606/272-2029
 APPOINTIVE OFFICER OF THE SOCIETY: DISK LIBRARIAN --- BOB MORRIS, Ph. 606/255-1572