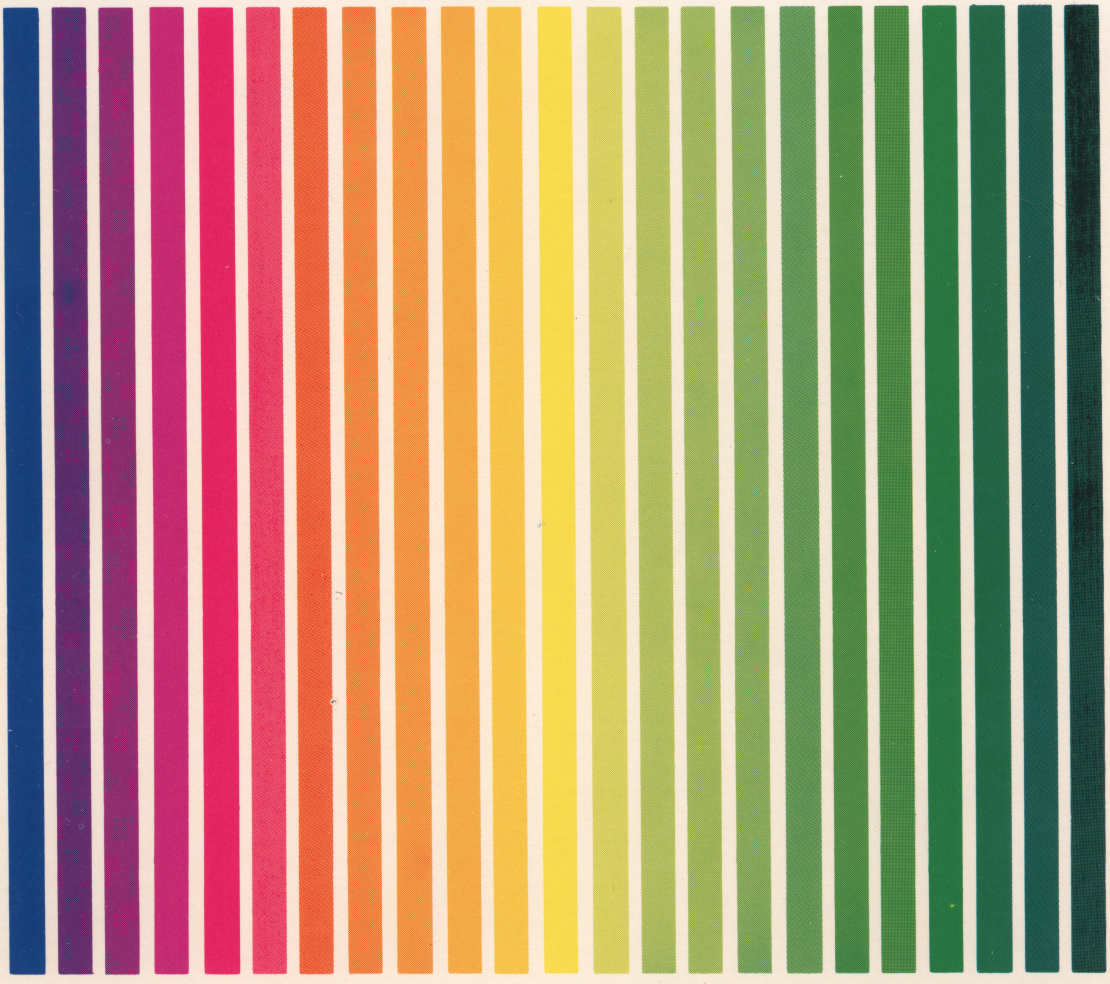
APX ATARI® PROGRAM EXCHANGE



Joseph J. Wrobel

T: A TEXT DISPLAY DEVICE

Intermix text and graphics on the
same line in any graphics mode

Cassette: 8K (APX-10067) Diskette: 16K (APX-20067)

User-Written Software for ATARI Home Computers

Joseph J. Wrobel

T: A TEXT DISPLAY DEVICE

Intermix text and graphics on the
same line in any graphics mode

Cassette: 8K (APX-10067) Diskette: 16K (APX-20067)

T: A TEXT DISPLAY DEVICE

by

Joseph J. Wrobel

TRADEMARKS OF ATARI

The following are trademarks of Atari, Inc.

ATARI®
ATARI 400™ Home Computer
ATARI 800™ Home Computer
ATARI 410™ Program Recorder
ATARI 810™ Disk Drive
ATARI 820™ 40-Column Printer
ATARI 822™ Thermal Printer
ATARI 825™ 80-Column Printer
ATARI 830™ Acoustic Modem
ATARI 850™ Interface Module

****

Distributed by

The ATARI Program Exchange
P. O. Box 427
155 Moffett Park Drive, B-1
Sunnyvale, CA 94086

To request an APX Software Catalog, write to the address above, or call toll-free:

800/538-1862 (outside California)
800/672-1850 (within California)

Or call our Sales number, 408/745-5535.

****

CONTENTS

INTRODUCTION

OVERVIEW

"T:" is an auto-loading, machine language routine that greatly expands the display capabilities of your ATARI Home Computer System. It's useful with text editing software, graphical presentation software (plot labeling is a snap), and any software where intermixing graphics and text is important. Using T:, you can intermix text and graphics freely on the same line on your TV screen without using modified display lists, PEEKs, or POKEs. The program defines a new device, T:, and uses it like P:. Printing to T: puts text on the screen; the text size is determined by your current graphics mode. This device resides with and is completely compatible with the ATARI computer's Operating System, DOS, and the ATARI BASIC Language Cartridge. You can use T: with any of the standard BASIC and OS graphics modes (both full and split screen), using text in as many colors as are available in that mode and selecting colors with the COLOR command.

T: prints the full ATASCII character set (both normal and inverse video) and also supports user-defined character sets. You can display full-width, mid-width, and half-width characters. The half-width mode results in an 80-by-24 line display in graphics mode 24 (8+16). T: supports both sequential line printing and random screen printing through the use of the POINT command. It recognizes two ATASCII control codes: the clear screen code (CHR$(125)) and the end-of-line code (CHR$(155)), which produces a screen carriage return/line feed.

T: has its own set of XIO commands that let you set left and right margins, select character width, alter the character base pointer for use with a user-defined character set, and offset text for such purposes as printing subscripts or superscripts or using proportional spacing.

REQUIRED ACCESSORIES

Cassette version
- 8K RAM
- ATARI 410 Program Recorder

Diskette version
- 16K RAM
- ATARI 810 Disk Drive

ATARI BASIC Language Cartridge

OPTIONAL ACCESSORIES

Monitor-quality display screen or black and white TV (for use with 80-by-24 character display mode)

CONTACTING THE AUTHOR

Users wishing to contact the author about T: may write to him at:

233 El Mar Drive
Rochester, NY 14616

GETTING STARTED

LOADING T: INTO COMPUTER MEMORY

If you have the cassette version of T:

1. Have your computer turned OFF.

2. Insert the T: cassette into the program recorder's cassette holder and press REWIND on the recorder until the tape rewinds completely. Then press PLAY to prepare the program recorder for loading the program.

3. Turn on the computer while holding down the START key.

4. When you hear a beep, release the START key and press the RETURN key. The program will load into computer memory.

If you have the diskette version of T:

1. Have your computer turned OFF.

2. Turn on your disk drive.

3. When the BUSY light goes out, open the disk drive door and insert the T: diskette with the label in the lower right-hand corner nearest to you. (Use disk drive one if you have more than one drive.)

4. Turn on your computer and your TV set. The program will load into computer memory.

FIRST DISPLAY SCREEN

After T: loads into computer memory, a large "T" displays in the center of the screen and the READY prompt appears beneath it.

INITIAL CONDITIONS

The handler to support the device "T:" now resides in computer memory (RAM), but it's safely tucked away where it won't interfere with normal machine operation or be lost if you press the SYSTEM RESET key. In disk systems, you won't lose T: when you call DOS if you have a MEM.SAV file on your diskette.

The only differences in the system are: (1) the device "T:" now exists, and (2) less user memory is available than with a normal system start-up. You can check memory size by typing ? FRE(0) and pressing the RETURN key. The number displayed is the number of free user bytes in RAM. This number is up to 510 bytes less than what would be available under a normal start-up. The character display routine of T: resides in this "stolen" RAM. Appendix I lists a full memory map of T:.

TECHNICAL DESCRIPTION

Character representation

T: divides the screen into a matrix of locations called cells. Each cell holds a single character. T: displays text by mapping the internal representation of a character into these cells bit-by-bit, pixel-by-pixel. Thus, in its normal, full-width mode (T cell width, TCW=8), T: transforms the eight-bit by eight-bit pattern in memory, which defines a character into an eight-pixel by eight-pixel display of that character on the screen. Half-width mode (TCW=4) uses an algorithm to squeeze the bit pattern into an eight-pixel high by four-pixel wide display. Likewise, the mid-width mode (TCW=5) uses an eight-pixel high by five-pixel wide cell. Because the character is drawn with fewer pixels in the reduced-width modes, it isn't as well resolved as it is in the full-width mode; however, each ATASCII character is distinct.

Cursor control

T: has its own invisible cursor, which resides in the cell in which the next T printed character is to display. Two variables determine the position of this cursor: TRC, the T row cursor, and TCC, the T column cursor. Like the graphics mode convention, T rows and columns are numbered starting with zero and with the origin at the upper left-hand corner of the screen.

The value of TRC can range from zero to one less than the maximum number of T rows, TMR. The extreme values TCC can assume are determined by the user-specified T left margin, TLM, and T right margin, TRM. The minimum value for TLM is zero; the maximum value for TRM is one less than the maximum number of T columns, TMC. An additional constraint on the margins is that TLM can't exceed TRM.

Number of characters that can fit on the screen

Two factors determine the number of characters T: can put on the screen: the current graphics mode and whether you're using full or reduced width characters. Table I lists the number of characters per line (TMC) and the number of lines per screen (TMR) T: supports for each of the available graphics modes. Note that in the character map display modes--ANTIC display modes 2 through 7--each "pixel" is actually a character of the standard size supported by that mode. Hence, T: forms very large characters out of smaller ones, and so the number of T characters displayable on the screen at a time is small. However, you can use these modes for short, attention-grabbing messages like PLAY WITH ME! or GO!. In the graphics display modes--ANTIC display modes 8 through F--each pixel is one color dot on the screen. In these modes, the pixel size determines the physical size of the character.

Offsets for fine control of character positioning

T: uses offsets to control character positioning for such purposes as using subscripts, superscripts, and proportional spacing. Two variables, TXO and TYO, maintain the values of the X and Y offsets. The variable's value indicates the number of pixels in the given direction a character is to be offset from the normal display position. Valid values of TYO are from zero to seven. TXO may range from zero to one less than the current T cell width.

TABLE I

CHARACTERS PER LINE (TMC) & LINES PER SCREEN (TMR)

ANTIC display modes (hex)	OS& BASIC graphic modes	Characters per line (TMC) full width	mid width	half width	Text lines per screen (TMR) split screen	full screen
2	0	5	8	10	-	3
3	(e)	5	8	10	2	2
4	(e)	5	8	10	2	3
5	(e)	5	8	10	1	1
6	1	2	4	5	2	3
7	2	2	4	5	1	1
8	3	5	8	10	2	4
9	4	10	16	20	5	6
A	5	10	16	20	5	6
B	6	20(a)	32	40	10	12
C	(e)	20(b)	32	40	20	24
D	7	20(a)	32	40	10	12
E	(e)	20(b)	32	40	20	24
F	8	40(c)	64	80(d)	20	24

(a) Character size equal to standard graphics mode 2 character.
(b) Character size equal to standard graphics mode 1 character.
(c) Character size equal to standard graphics mode 0 character.
(d) Requires monochrome display for good visibility.
(e) Use of these ANTIC display modes requires display list modification.

COMMANDS

INTRODUCTION

This section describes all the ATARI BASIC commands used to operate T:. None is new; all are described fully in the ATARI BASIC Reference Manual (abbreviated in these discussions as "ABRM"), in either Section 5, "Input/Output Commands and Devices", or in Section 9, "Graphics Modes and Commands". For your convenience, each command title is followed by a bracketed reference to the ABRM section and page describing the command.

OPEN [Sec. 5, p. 26]

```
Format:  OPEN #aexp,aexp1,0,"T:"
Example: OPEN #1,8,0,"T:"
```

As with any other device, you must open T before you access it. The allowable parameters for the OPEN command are:

aexp = Reference IOCB. See ABRM for detailed discussion.
aexp1 = Operation code.

T allows either:

aexp1 = 8, open for output and initialize
aexp1 = 9, open for output and retain current configuration.

If the operation code is set to 8, the following variables are set to the given initial values:

Variable		Description	Initial value
TCB	T	character base	224
TCC	T	column cursor	0
TCW	T	cell width	8 (full width)
TLM	T	left margin	0
TRC	T	row cursor	0
TRM	T	right margin	79
TXO	T	X offset	0
TYO	T	Y offset	0

Note that OPENing T: doesn't affect the display. Moreover, you can change the graphics mode at will both before and after OPENing T:. T: always adapts to the current graphics mode by recalculating the TMC and TMR whenever it's called.

CLOSE [Sec. 5, p. 26]
Format: CLOSE #aexp
Example: CLOSE #1

aexp = Reference IOCB from OPEN command. See ABRM for an explanation of this command. Note that like the OPEN command, CLOSE doesn't affect the display.

POINT [Sec. 5, p. 28]
Format: POINT #aexp,avar,avar
Example: POINT #1,X,Y

aexp = Reference IOCB from OPEN command. Use this command to place the invisible T cursor at a specific location on the screen. The first avar specifices the T column (TCC) and the second avar specifies the T row (TRC). In this way, using the POINT command with T is like using the POSITION command with graphics.

The next character printed by T: appears at the specified cell, unless either the T cursor is positioned outside the current T margins or TRC exceeds TMR. In the former case, the next T character will display at the left margin either in the current row (if TCC is less than TLM) or in the next row (if TCC is set greater than TRM). In the latter case, the screen will scroll, then display the next T character at the left margin of the bottom row.

NOTE [Sec. 5, p. 26]
Format: NOTE #aexp,avar,avar
Example: NOTE #1,X,Y

aexp = Reference IOCB from OPEN command. Use this command to obtain the position of the invisible T cursor. The current values of TCC and TRC are returned in the first and second avar.

PRINT [Sec. 5, p. 28]
Format: PRINT #aexp[{;,}exp...][;]
Example: PRINT #1;"Hello",X^2;A$

aexp = Reference IOCB from OPEN command. This command causes the information contained in the string or numeric expressions to display on the screen, starting at the current T cursor position. Line overflow results in an automatic carriage return/line feed (CRLF). Screen overflow results in automatic scrolling. If you don't use a comma or semicolon at the end of the PRINT statement, then a CRLF is generated and the next PRINT statement starts displaying text at the left margin on the following line. Only two control characters are recognized as such: CHR$(125) causes the whole screen to clear; CHR$(155) generates a CRLF.

```
PUT [Sec. 5, p. 28]
      Format:  PUT #aexp,aexp1
      Example: PUT #1,56
```

aexp = Reference IOCB from OPEN command. The PUT command outputs a single byte from 0 to 255, which T: interprets as an ATASCII character to be displayed. This command is equivalent to PRINT #aexp;CHR$(aexp1); .

```
COLOR [Sec. 9, p. 48]
      Format:  COLOR aexp1
      Example: COLOR 1
```

The operation and action of the COLOR command are the same as described in the ABRM, as is true for all the graphics commands. The COLOR command, however, affects the operation of T:. When T: displays a character, the background (foreground for inverse characters) is always plotted using COLOR 0. The color of the foreground (background for inverse characters), however, is determined by the value of aexp1 in the last executed COLOR command in the same way that PLOT and DRAWTO commands are affected. Thus, in the nontext graphics modes, the COLOR command determines the colors of the characters T: displays. In the text graphics modes, as stated earlier, the characters T: displays are composed of "pixels", which are themselves characters. In this case, the COLOR command determines what these "building blocks" characters are to be.

```
STATUS [Sec. 5, p. 29]
      Format:  STATUS #aexp,avar
      Example: STATUS #1,S
```

aexp = Reference IOCB from OPEN command. The STATUS command returns to avar the value of TCW, the current T cell width. Valid values are either 4, 5, or 8.

```
XIO [Sec. 5, p. 29]
      Format:  XIO cmdno,#aexp,aexp1,aexp2,"T:"
      Example: XIO 100,#1,2,18,"T:"
```

T: supports its own set of general I/O commands. The parameters for these commands are used as follows:

cmdno	Operation	Parameter use
100	set margins	TLM=aexp1, TRM=aexp2
101	set cell width	TCW=aexp1, aexp2 ignored
102	set offsets	TXO=aexp1, TYO=aexp2
103	set character base	TCB=aexp1, aexp2 ignored

Acceptable values for TLM, TRM, TCW, TXO, and TYO are given in the section titled "Technical Description".

TCB is a character set pointer used only by T:. Initially, it points to the ROM-based ATASCII character set. However, using the XIO 103 command, you can alter it to point to a

user-defined character set. In this way, you can set up T: to use one character set while normal screen print operations use another.

PROGRAMMING HINTS

BLANK SCREENS

Don't forget color. If repeated attempts at displaying characters through T: leave you with a blank screen even though all else looks all right, check the current color. Remember, a character's background is always displayed in COLOR 0. If the COLOR statement isn't executed in your program, it may be set to its default value of zero. This means the foreground will display in the same color as the background, making characters quite difficult to see!

OFFSETS

The X and Y offsets can take on only positive values. But what if you want a negative Y offset for superscripts, for example? Use NOTE and POINT to place the T cursor on the line directly above where you want the superscript to occur, and then use a positive Y offset to lower it to the proper position. To display the next non-superscripted character, use POINT to set the cursor back on line and reset the Y offset to zero. The same approach applies to getting negative X offsets.

VERTICAL PRINTING MADE EASY

A simple trick lets you do vertical printing easily with T:. First, position the T cursor at the first character position. Now set both the left and right T margins to the same column position, and PRINT. That's all there is to it! Readjust the margins to resume normal printing.

MIXING TEXT AND GRAPHICS

Because T: allows text and graphics to share the display simultaneously, which should be done first? Order does affect the final result, because when T: displays a character, it colors in both the foreground and the background. Hence, if a character displays over part of a graphics display, the character will obliterate a TCW-by-eight pixel area. On the other hand, if the text displays first, you risk running lines right through it, making deciphering the text difficult. The solution is to put the text where the graphics aren't. If you can't, then give the advantage to the text if it's important; if it isn't important, leave it out.

WIDTH AFFECTS POSITION

Changing cell width in the middle of a line of text may cause you to put characters where you don't want them. This displacement occurs because when the width changes, the column cursor isn't changed but is rescaled. For example, suppose you're in GRAPHICS 8 at column 20. If you're printing full-width characters, you're right in the middle of the line (Table I shows a line of GRAPHICS 8 holds up to 40 full-width characters). But now you switch to half-width characters for a special effect. The column cursor is still set at

20, but where will the next character display? Well, Table I says 80 half-width characters can fit in a line of GRAPHICS 8. Thus, column 20 is now only a quarter of the way across the screen! Use NOTE and POINT to readjust the T column cursor when you change character width mid-line.

APPENDIX A

T: Memory Map

T uses three blocks in memory. The first block resides on page zero in hexadecimal locations $E0 through $E6. The variables stored in these locations are only temporary. Thus, T: will modify these locations but other programs can use them between T: calls.

The second block occupies page six from hexadecimal locations $06A7 through $06FF. This block stores permanent variables, the device handler table for T:, and the initialization routine. The hexadecimal locations of the permanent variables are as follows:

TRC	$06A7	TXO	$06A9	TLM	$06AB	TCB	$06AF
TCC	$06A8	TYO	$06AA	TRM	$06AC	TCW	$06B0

The third block contains all the handler routines. The absolute location of this relocatable code depends on your system configuration. In a cassette-based system, the code always originates at hexadecimal location $0700. In a disk-based system without an ATARI 850 Interface Module turned on, the routines always occupy the 510 bytes just preceding the OS MEMLO. You can calculate the decimal address of OS MEMLO by the following expression:

```
PEEK(743)+256*PEEK(744)
```

APPENDIX B:

T: Command Summary

CLOSE #aexp --> Closes "T:"; no effect on display

COLOR aexp1 --> Sets character color

NOTE #aexp,avar,avar --> Return current T cursor position (X,Y)

OPEN #aexp,8,0,"T:" --> Opens and initializes T

OPEN #aexp,9,0,"T:" --> Opens T without initialization

POINT #aexp,avar,avar --> Positions the invisible T cursor (X,Y)

PRINT #aexp[{;,} exp...][{;,}] --> Passes expressions to T for display

PUT #aexp,aexp1 --> Causes T to display a single character

STATUS #aexp,avar --> Returns current T cell width (4, 5, or 8)

XIO 100,#aexp,aexp1,aexp2,"T:" --> Sets left and right T margins

XIO 101,#aexp,aexp1,aexp2,"T:" --> Sets T cell width (4, 5, or 8)

XIO 102,#aexp,aexp1,aexp2,"T:" --> Sets X and Y offsets

XIO 103,"aexp,aexp1,aexp2,"T:" --> Sets T character base

APPENDIX C

Demonstration Programs

```
1 REM Text Mode Demo for T
10 GRAPHICS 0
20 ? "Which graphics text mode"
30 ? "  demonstration would you"
40 ? "    like to see?":?
50 ? "        Key 0, 1 or 2. ";
60 OPEN #1,4,0,"K:"
70 GET #1,G:G=G-48:IF G<0 OR G>2 THEN 70
80 GRAPHICS 0
90 ? "Which cell width would"
100 ? "  you like to see?":?
110 ? "      Key 4, 5 or 8. ";
120 GET #1,W:W=W-48:IF W<>4 AND W<>5 AND W<>8 THEN 120
130 CLOSE #1
140 GRAPHICS G+16:POSITION 0,23
150 OPEN #1,8,0,"T:"
160 XIO 101,#1,W,0,"T:"
170 FOR I=0 TO 255
180 IF I=125 OR I=155 THEN PUT #1,32:GOTO 220
190 C=I-128*(I>128):C=C+64-96*(C>31)+32*(C>95)+128*(I>128):COLOR C
200 PUT #1,I
210 FOR J=1 TO 20+G*40:NEXT J
220 NEXT I
230 GOTO 170
240 END
```

```
1 REM Two Color Graphics Demo for T
10 GRAPHICS 0
20 ? "Which two color graphics"
30 ? "  mode demonstration would"
40 ? "    you like to see?":?
50 ? "        Key 4, 6 or 8. ";
60 OPEN #1,4,0,"K:"
70 GET #1,G:G=G-48:IF G<>4 AND G<>6 AND G<>8 THEN 70
80 GRAPHICS 0
90 ? "Which cell width would"
100 ? "  you like to see?":?
110 ? "      Key 4, 5 or 8. ";
120 GET #1,W:W=W-48:IF W<>4 AND W<>5 AND W<>8 THEN 120
130 CLOSE #1
140 OPEN #1,8,0,"T:"
150 GRAPHICS G+16
160 COLOR 1
170 XIO 101,#1,W,0,"T:"
180 FOR I=0 TO 255
190 IF I=125 OR I=155 THEN PUT #1,32:GOTO 210
200 PUT #1,I
210 NEXT I
220 GOTO 180
230 END
```

```
1 REM Four Color Graphics Demo for T
10 GRAPHICS 0
20 ? "Which four color graphics"
30 ? "  mode demonstration would"
40 ? "    you like to see?":?
50 ? "        Key 3, 5 or 7. ";
60 OPEN #1,4,0,"K:"
70 GET #1,G:G=G-48:IF G<>3 AND G<>5 AND G<>7 THEN 70
80 GRAPHICS 0
90 ? "Which cell width would"
100 ? "  you like to see?":?
110 ? "      Key 4, 5 or 8. ";
120 GET #1,W:W=W-48:IF W<>4 AND W<>5 AND W<>8 THEN 120
130 CLOSE #1
140 OPEN #1,8,0,"T:"
150 GRAPHICS G+16
160 C=0
170 XIO 101,#1,W,0,"T:"
180 FOR I=0 TO 255
190 C=C+1:IF C=4 THEN C=1
200 COLOR C
210 IF I=125 OR I=155 THEN PUT #1,32:GOTO 230
220 PUT #1,I
230 NEXT I
240 GOTO 180
250 END
```

```
1 REM Reduced Width Demo for T
10 DIM CHARSET$(256):? :? "WORKING..."
20 FOR I=1 TO 256
30 CHARSET$(I,I)=CHR$(I-1)
40 IF I=126 OR I=156 THEN CHARSET$(I,I)=" "
50 NEXT I
60 GRAPHICS 24
70 COLOR 1
80 OPEN #1,8,0,"T:"
90 DATA 8,4,35,1,5,0,63,2,4,8,71,2
100 FOR I=1 TO 3:READ TCW,TLM,TRM,N
110 XIO 100,#1,TLM,TRM,"T:"
120 XIO 101,#1,TCW,0,"T:"
130 NOTE #1,X,Y:POINT #1,TLM,Y
140 FOR J=1 TO N
150 PRINT #1;CHARSET$;
160 NEXT J
170 IF I<3 THEN PRINT #1
180 NEXT I
190 GOTO 190
200 END
```

```
1 REM Superscript Demo for "T:"
2 REM Prints inverse mode characters
3 REM as superscripted characters.
10 OPEN #1,8,0,"T:":DIM A$(40)
20 GRAPHICS 8:COLOR 1
30 A$="X3-1 = (X-1)*(X2+X+1)"
40 X=10:Y=10:POINT #1,X,Y
50 FOR I=1 TO LEN(A$)
60 Z=ASC(A$(I,I))
70 IF Z<128 THEN PUT #1,Z:GOTO 130
80 NOTE #1,X,Y:Y=Y-1:POINT #1,X,Y
90 XIO 102,#1,0,4,"T:"
100 PUT #1,Z-128
110 XIO 102,#1,0,0,"T:"
120 NOTE #1,X,Y:Y=Y+1:POINT #1,X,Y
130 NEXT I
140 END
```

```
1 REM Text/Graphics Demo for "T:"
10 OPEN #1,8,0,"T:"
20 GRAPHICS 8:COLOR 1
30 X=8:Y=1:POINT #1,X,Y
40 ? #1;"MIXING TEXT AND GRAPHICS:"
50 X=17:Y=Y+2:POINT #1,X,Y
60 ? #1;"  as  "
70 X=17:Y=Y+2:POINT #1,X,Y
80 ? #1;"simple"
90 X=17:Y=Y+2:POINT #1,X,Y
100 ? #1;"  as  "
110 PLOT 10,40:DRAWTO 70,90
120 DRAWTO 100,40:DRAWTO 10,40
130 X=8:Y=7:POINT #1,X,Y
140 PUT #1,ASC("A")
150 PLOT 200,50:DRAWTO 270,50
160 DRAWTO 270,100:DRAWTO 200,100
170 DRAWTO 200,50
180 X=29:Y=9:POINT #1,X,Y
190 PUT #1,ASC("B")
200 Z=20*ATN(1)/9:PLOT 170,116
210 FOR I=0 TO 18
220 X=140+30*COS(I*Z)
230 Y=116+30*SIN(I*Z)
240 DRAWTO X,Y:NEXT I
250 X=17:Y=14:POINT #1,X,Y
260 PUT #1,ASC("C")
270 X=25:Y=14:POINT #1,X,Y
280 XIO 100,#1,25,25,"T:"
290 ? #1;"with";
300 XIO 100,#1,0,39,"T:"
310 X=27:Y=17:POINT #1,X,Y
320 ? #1;CHR$(34);"T:";CHR$(34);"!"
330 END
```

```
1 REM Subscript Demo for "T:"
2 REM Prints inverse mode characters
3 REM as subscripted characters.
10 OPEN #1,8,0,"T:":DIM A$(40)
20 GRAPHICS 8:COLOR 1
30 A$="2 H2 + 02   2 H20"
40 X=10:Y=10:POINT #1,X,Y
50 FOR I=1 TO LEN(A$)
60 Z=ASC(A$(I,I))
70 IF Z<128 THEN PUT #1,Z:GOTO 110
80 XIO 102,#1,0,4,"T:"
90 PUT #1,Z-128
100 XIO 102,#1,0,0,"T:"
110 NEXT I
120 END
```

LIMITED WARRANTY ON MEDIA AND HARDWARE ACCESSORIES.

We, Atari, Inc., guarantee to you, the original retail purchaser, that the medium on which the APX program is recorded and any hardware accessories sold by APX are free from defects for thirty days from the date of purchase. Any applicable implied warranties, including warranties of merchantability and fitness for a particular purpose, are also limited to thirty days from the date of purchase. Some states don't allow limitations on a warranty's period, so this limitation might not apply to you. If you discover such a defect within the thirty-day period, call APX for a Return Authorization Number, and then return the product along with proof of purchase date to APX. We will repair or replace the product at our option.

You void this warranty if the APX product: (1) has been misused or shows signs of excessive wear; (2) has been damaged by use with non-ATARI products; or (3) has been serviced or modified by anyone other than an Authorized ATARI Service Center. Incidental and consequential damages are not covered by this warranty or by any implied warranty. Some states don't allow exclusion of incidental or consequential damages, so this exclusion might not apply to you.

DISCLAIMER OF WARRANTY AND LIABILITY ON COMPUTER PROGRAMS.

Most APX programs have been written by people not employed by Atari, Inc. The programs we select for APX offer something of value that we want to make available to ATARI Home Computer owners. To offer these programs to the widest number of people economically, we don't put APX products through rigorous testing. Therefore, APX produts are sold "as is", and we do not guarantee them in any way. In particular, we make no warranty, express or implied, including warranties of merchantability and fitness for a particular purpose. We are not liable for any losses or damages of any kind that result from use of an APX product.

ATARI PROGRAM EXCHANGE

REVIEW FORM

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many software authors are willing and eager to improve their programs if they know what users want. And, of course, we want to know about any bugs that slipped by us, so that the software author can fix them. We also want to know whether our documentation is meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program _____

2. If you have problems using the program, please describe them here.

_____

_____

_____

3. What do you especially like about this program?

_____

_____

_____

4. What do you think the program's weaknesses are?

_____

_____

_____

5. How can the catalog description be more accurate and/or comprehensive?

_____

_____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program?

_____ Easy to use
_____ User-oriented (e.g., menus, prompts, clear language)
_____ Enjoyable
_____ Self-instructive
_____ Useful (non-game software)
_____ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

8. What did you especially like about the user instructions?

9. What revisions or additions would improve these instructions?

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

11. Other comments about the software or user instructions:

STAMP

ATARI Program Exchange
Attn: Publications Dept.
P.O. Box 50047
60 E. Plumeria Drive
San Jose, CA 95150

[seal here]