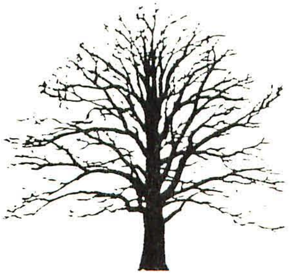
SoftSide™ Selections

#42

SCREAMIN' DEMON

Microtext 2.0

Family Tree Organizer



The Arabian Nights

A product of **SoftSide** Publications, Inc.
10 Northern Blvd., Amherst, NH 03031

SoftSide™ Selections

Index

This booklet prepared by the staff of SoftSide Publications, Inc. Entire contents copyright © 1983 by SoftSide Publications, Inc., 10 Northern Blvd., Northwood Executive Park, Amherst, NH 03031.

by Jon Voskuil

Modifications and enhancements by the **SoftSide** programming staff.

Microtext 2.0 is a word processor for a 48K Atari®.

Upon running, *Microtext 2.0* displays a mostly blank screen with an instruction summary line at the top or bottom. You can either start typing, or load a previously saved file from disk or tape.

You will use the CTRL key to access editor and system functions. Holding down CTRL, and then pressing S, L, R, P, or E will access the save, load, review, printout, or edit functions. Although not mentioned in the command summary on the screen, pressing CTRL-Q will quit the program.

Saving and loading files is simply a matter of answering the questions about the medium to be used (tape or disk) and, if disk, the file name. Once you have entered a file name, it will be used as the default until you specify another one or exit the program: just press Return when asked for the file name. This simplifies repeated saves during entry of a long document.

The review function causes the computer to return to the beginning of the text in memory and scroll through it to the end. During this scrolling, you can press the space bar to pause. Then, pressing the space bar will cause one or more lines to be displayed; pressing Return will cause the scrolling to continue; and pressing CTRL-E will enter the editing mode; ESC cancels review.

In the editing mode, you can move the cursor up and down through your text, to locate any line which you want to edit or delete. This movement is accomplished with the up- and down-arrow keys (CTRL must be held down). You have four options while in the editing mode: pressing ESC will exit to the review mode; pressing CTRL-D will delete the line at the cursor; pressing *CTRL-X will* delete everything from the cursor to the end of the text; pressing Return will allow you to edit the line at the cursor; and pressing CTRL-F will let you find a string in your text.

If you choose to edit a line, the screen will clear, and then display a number of lines of text with a gap of several lines in the middle. The cursor will be positioned at the beginning of the line you have chosen to edit, and you can proceed to type in a new line to replace the old one. The new one can be shorter than the original, or may occupy multiple screen lines. Any part of the original line that you want retained must be retyped: whatever you type in will replace the entire line. You can use the right-arrow key (with CTRL) to retype automatically the portion of the line you wish to retain. When you have finished entering the new text, press CTRL-F (not return, unless you want a carriage return in the text itself). The computer will check to see if the text lines need to be rearranged, and then return you to the review mode.

The find function, accessed by pressing CTRL-F from the edit mode, will ask you what string to search for. The program will search through your text for your string, beginning with the line the cursor is on, and put the cursor on the first line in which your string was found. Since a multiple-word search string may be broken up between lines, it is more reliable to search for single words.

The find routine will leave you editing the line in which it found your search string; if no change is desired, simply copy the line with the right-arrow. The find routine displays a message to inform you if the string was not found.

The printout function (CTRL-P) allows you to send your text to a printer, after selecting margins, and line spacing.

Variables

AZ: General use.
AZ$: String to be searched for.
B$: Contains the backspace character [CHR$(126)].
BKSP: Equals 126, the ASCII code of the backspace character.
C: The ASCII code of the character the user most recently typed.
C$: The actual most recently typed character or its equivalent.
C1: A temporary storage variable used to preserve the value of CHAR during the editing of a line.
CC: The ASCII coue of a character.
CH: Pointer used in the rejustification routine.
CHAR: Character position within the line currently being typed.
CL$: Clear-screen character.
CN: Character pointer in the edit routine.
CR: A flag that recalls whether a line ended in a carriage return.
CR$: Contains the character used to represent a carriage return on the screen.
E: Error code.
ED: Flag that recalls whether a line is currently being edited.
EL: Number of the line currently being edited.
FL: If FL = 1, then there is a partial line at the end of the text. Used in the edit routine.
F$: File name.
F1$: Previous file name (if any).
FT$: Temporary file name.
H: Horizontal cursor position.
I: General loop variable.
IT: Used in screen display routines while editing a line.
J: General loop variable.
K: Pointer used in the edit routine.
L: Temporary variable.
L$: Text in current line.
L1: Used in edit routines.
L2: Used in edit routines.

MICROTEXT 2.0

LIN: Temporarily holds a line number for the edit and review routines.
LIN$: Contains a line of hyphens.
LL: Line length.
LM: Left margin.
LN: Number of lines in the array.
LNXT$: Temporarily holds the next line line of text.
LP: Length of text to be printed.
LP(*): Line pointers to lines in T$.
LS: Line spacing.
LWID: Maximum number of characters per line.
N: Used to rejustify text.
NC: Next character pointer in the edit routine.
NL: Pointer used in rejustification routine.
NN: Pointer used to display lines in the edit routine.
P$: Used to set up text to be printed.
PF: Print flag used for error trapping.
PP$: Text to be printed.
Q$: Contains the prompt lines.
RM: Right margin.
RTN: ASCII code of the Return key.
S$: A line of spaces.
S1: Temporary storage for SLOC.
SL: Same as SLOC.
SLOC: Last character position at which a space occurred.
SS: Number of characters to erase.

SPC: ASCII code of the space character.
STP: Flag used in the review mode that determines whether to step through line by line.
TL: Holds the text length.
TLN: Used to manage string position in the edit routine.
T$: Main string to hold text.
TT$: Used to load text from media.
V: Vertical cursor position.
V1: Vertical screen position.
VV: Vertical cursor position.
X: General use.
X$: General use.
Z: Timing loop variable.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                            SS
SS         Atari BASIC        SS
SS       'Microtext 2.0'      SS
SS   Author: Jon R. Voskuil   SS
SS      Copyright © 1983      SS
SS SoftSide Publications, Inc SS
SS                            SS
SS SS SS SS SS SS SS SS SS SS SS
```

If you don't wish to type this program, it is available on issue #42 SoftSide CV and DV.

Title page.

```
10 GRAPHICS 0
20 POKE 752,1:POKE 709,2:POKE 710,12
30 POSITION 13,8:? " MICROTEXT 2.0 ":P
OSITION 8,10:? "WRITTEN BY JON R. VOSK
UIL":POSITION 6,12:COLOR 32
40 ? "MODIFICATIONS BY ALAN J. ZETT":?
 :? "COPYRIGHT 1983, SOFTSIDE PUBLICAT
IONS":FOR Z=1 TO 1000:NEXT Z
```

Initialization.

```
100 TRAP 20000
120 DIM L$(40),T$(14000),B$(1),C$(1),C
R$(1),LIN$(37),LNXT$(40),F$(14),S$(37)
,FT$(14),LP(500),TT$(40),X$(5),Q$(37)
125 DIM P$(255),PP$(80),F1$(14),CL$(1)
,AZ$(25):CL$=CHR$(125)
130 BKSP=126:RTN=155:SPC=32:PF=0:B$=CH
R$(BKSP):CR$=CHR$(20)
135 T$="":L$="":LP(0)=0
140 CHAR=1
150 LN=1
160 LIN$="----------------------------
---------":S$=" ":S$(37)=" ":S$(2)=S$(
1):? CL$
170 OPEN #1,4,0,"K:"
180 LWID=36
190 V=2:H=2
200 Q$="^SAVE  ^LOAD  ^REVIEW  ^EDIT
^PRINT "
210 POKE 752,1:POSITION 2,0:? Q$:? LIN
$:POSITION H,V:POKE 752,0:? " ";B$;
```

Input loop.

```
500 GET #1,C:C$=CHR$(C)
510 IF C=31 AND ED THEN IF CHAR<(LP(EL
)-LP(EL-1)+1) THEN C$=T$(CHAR+LP(EL-1)
):C=ASC(C$):IF C=20 THEN C=155
520 IF C=RTN THEN C$=CR$
640 IF C<>BKSP THEN 720
650 IF CHAR<2 THEN L$="":GOTO 500
660 ? B$;" ";B$;
670 CHAR=CHAR-1
675 IF CHAR<2 THEN 500
680 L$=L$(1,CHAR-1)
700 GOTO 500
720 IF C<>RTN THEN IF C<SPC OR C>250 O
R (C>122 AND C<160) THEN 2000
740 CHAR=CHAR+1:IF CHAR<LWID OR C=SPC
OR C=RTN THEN 760
750 GOSUB 1000
760 L$(CHAR-1)=C$
780 IF C<>RTN THEN 880
820 ? C$;S$(1,38-PEEK(85))
840 GOSUB 6000:SLOC=0
850 CHAR=1
860 GOTO 500
880 ? C$;
900 IF C=SPC THEN SLOC=CHAR-1:IF CHAR=
LWID THEN GOSUB 6000:CHAR=1:SLOC=0:?
920 GOTO 500
```

Subroutine to break line at a space, and to initialize the next line.

```
1000 IF SLOC=0 THEN ? :GOTO 1100
1020 SS=LWID-SLOC-1:FOR J=1 TO SS:? B$
;:NEXT J
1040 FOR J=1 TO SS:? " ";:NEXT J:?
1050 IF SLOC=LEN(L$) THEN LNXT$="":GOT
O 1100
1060 LNXT$=L$(SLOC+1)
1080 L$=L$(1,SLOC)
1100 GOSUB 6000
1110 L$=LNXT$
1115 LNXT$=""
1120 ? L$;
1140 CHAR=LEN(L$)+2
1150 SLOC=0
1160 RETURN
```

Subroutine to process command codes.

```
2000 V=PEEK(84):H=PEEK(85):POSITION H+
1,V:POKE 752,1:? CHR$(126);
2050 TL=LEN(T$):T$(TL+1)=L$
2060 PLOT 0,0:DRAWTO 39,0:PLOT 0,1
2070 DRAWTO 39,1:POSITION 2,1:? LIN$
2080 IF ED=0 OR C<>6 THEN 2100
2085 IF CHAR>1 THEN LP(LN)=LEN(T$):LN=
LN+1:LP(LN)=LEN(T$):L$=""
2090 ED=0:RETURN
2100 IF C=18 AND ED=0 THEN GOSUB 3000
2200 IF C=19 AND ED=0 THEN GOSUB 4000
2300 IF C=12 AND ED=0 THEN GOSUB 5000
2400 IF C=17 THEN GRAPHICS 0:END
2500 IF C=16 AND ED=0 THEN GOSUB 7000
2600 IF C=5 AND ED=0 AND LN>1 THEN I=L
N-1:VV=V:POSITION 2,VV-1:GOSUB 9000:GO
SUB 3000
2900 IF TL>0 THEN T$=T$(1,TL)
2950 GOTO 200+10*ED
```

Subroutine to review entered text.

```
3000 ? CL$;"Press any key to pause":?
LIN$
3040 IF LN=1 THEN 3210
3050 FOR I=1 TO LN-1
3060 ? T$(LP(I-1)+1,LP(I))
3070 IF PEEK(764)=255 AND  NOT STP THE
N 3200
3080 STP=0:POKE 764,255
3090 VV=PEEK(84)
3100 POSITION 2,0:? " RTN:Cont  SPC:St
p  E:Edit  ESC:Exit ":? LIN$;
3120 X=PEEK(764):POKE 764,255
3125 IF X=170 THEN GOSUB 9000:GOTO 300
0
3130 IF X<>28 THEN 3160
3135 IF LN-I<22 THEN 3190
3140 X=LN-21:IF X<1 THEN X=1
3150 ? CL$:? LIN$:FOR AZ=X TO LN-1:? T
$(LP(AZ-1)+1,LP(AZ)):NEXT AZ:V=PEEK(84
):H=PEEK(85):RETURN
3160 IF X=12 THEN 3190
3170 IF X<>33 THEN 3120
3180 STP=1
3190 POSITION 2,VV:POKE 764,255
3200 NEXT I
3210 X=LP(LN-1)+1:IF X<=LEN(T$) THEN ?
 T$(X);
3220 H=PEEK(85):V=PEEK(84)
3230 RETURN
```

Subroutine to save text on tape or disk.

```
4000 PLOT 0,0:DRAWTO 39,0:POSITION 2,0
:? "Save to Tape or Disk? (T/D/ESC)";
4020 GET #1,X:X=X-128*(X>127)
4030 X=X-32*(X>90):IF X=27 THEN 4400
4060 IF X=84 THEN 4200
4070 IF X<>68 THEN 4000
4075 F1$=F$
4080 PLOT 0,0:DRAWTO 39,0:POSITION 2,0
:? "File Name: ";:INPUT F$
4082 IF F$="" AND F1$="" THEN 4080
4083 IF F$="" THEN F$=F1$
4085 IF F$(2,2)<>":" AND F$(3,3)<>":"
THEN FT$="D:":FT$(3)=F$:F$=FT$
4090 POSITION 2,0:? "Insert disk and p
ress RETURN";:GET #1,X:GOSUB 10000
4100 OPEN #2,8,0,F$:PLOT 0,0:DRAWTO 39
,0:POSITION 2,0:? "SAVING '";F$;"'":GO
TO 4210
4200 POSITION 2,0:? "Start tape record
er and press RETURN";:GET #1,X
4205 OPEN #2,8,0,"C:"
4210 ? #2;LN:? #2;SLOC:? #2;CHAR
4220 FOR I=1 TO LN-1
4230 ? #2;T$(LP(I-1)+1,LP(I))
4240 NEXT I:IF CHAR>1 THEN ? #2;T$(LP(
LN-1)+1)
4300 CLOSE #2
4400 RETURN
```

Subroutine to load text from tape or disk.

```
5000 PLOT 0,0:DRAWTO 39,0:POSITION 2,0
:? "Load from Tape or Disk? (T/D/ESC)"
;:GET #1,X:X=X-128*(X>127)
5030 X=X-32*(X>90):IF X=27 THEN 5400
5060 IF X=84 THEN 5200
5070 IF X<>68 THEN 5000
5075 F1$=F$
5080 PLOT 0,0:DRAWTO 39,0:POSITION 2,0
:? "File Name: ";:INPUT F$
5082 IF F$="" AND F1$="" THEN 5080
5083 IF F$="" THEN F$=F1$
5085 IF F$(2,2)<>":" AND F$(3,3)<>":"
THEN FT$="D:":FT$(3)=F$:F$=FT$
5090 POSITION 2,0:? "Insert disk and p
ress RETURN";:GET #1,X:GOSUB 10000
5100 OPEN #2,4,0,F$:PLOT 0,0:DRAWTO 39
,0:POSITION 2,0:? "LOADING '";F$;"'":G
OTO 5210
5200 POSITION 2,0:? "Start tape record
er and press RETURN";:GET #1,X
5205 OPEN #2,4,0,"C:"
5210 INPUT #2;LN:INPUT #2;SLOC
5215 INPUT #2;CHAR:T$=""
5220 FOR I=1 TO LN-1
5230 INPUT #2,TT$:T$(LEN(T$)+1)=TT$
5240 LP(I)=LEN(T$)
5250 NEXT I
5255 TL=LEN(T$):L$=""
5260 IF CHAR>1 THEN INPUT #2,TT$:T$(LE
N(T$)+1)=TT$:LP(LN)=LEN(T$):L$=TT$
5300 CLOSE #2
5350 GOSUB 3000
5400 RETURN
```

Subroutine to add a line of text to the main text string.

```
6000 T$(LEN(T$)+1)=L$
6080 LP(LN)=LEN(T$)
6100 LN=LN+1:LP(LN)=LEN(T$)
6150 L$=""
6200 RETURN
```

Subroutine to print the text in memory on a printer.

```
7000 PF=1:? CL$:POSITION 2,6:LIN=0:? "
Left margin? (Default = 10) ";
7010 INPUT X$:LM=10:IF LEN(X$)>0 THEN
IF VAL(X$)>0 THEN LM=VAL(X$):IF LM>37
THEN LM=37
7020 ? :? "Right margin? (Default = 70
) ";:INPUT X$:RM=70:IF LEN(X$)>0 THEN
IF VAL(X$)>0 THEN RM=VAL(X$)
7030 ? :? "Line spacing? (Default = 2)
 ";:INPUT X$:LS=2:IF LEN(X$)>0 THEN IF
 VAL(X$)>0 THEN LS=VAL(X$)
7040 LL=RM-LM
7070 ? CL$:LPRINT "":P$="":CR=0:I=0
7080 I=I+1:P$(LEN(P$)+1)=T$(LP(I-1)+1,
LP(I))
7090 IF P$(LEN(P$))=CR$ THEN CR=1:GOTO
 7110
7100 IF LEN(P$)<255-LWID AND I<LN-1 TH
EN 7080
7110 GOSUB 7500:CR=0
7120 IF I<LN-1 THEN 7080
```

```
7130 LPRINT S$(1,LM);L$;
7150 LPRINT ""
7160 GOSUB 3000
7170 RETURN
7500 L=LL
7510 IF LEN(P$)>LL THEN 7550
7520 IF ( NOT CR) THEN 7640
7530 LP=LEN(P$):IF LP<2 THEN PP$="":P$
="":GOTO 7590
7540 PP$=P$(1,LP-1):P$="":GOTO 7590
7550 C$=P$(L,L):IF C$=" " THEN 7580
7560 L=L-1:IF L>0 THEN 7550
7570 L=LL
7580 PP$=P$(1,L):P$=P$(L+1)
7590 LPRINT S$(1,LM);PP$;
7610 FOR J=1 TO LS:LIN=LIN+1:LPRINT ""
:NEXT J
7615 IF LIN>59 THEN FOR J=1 TO 66-LIN:
LPRINT "":NEXT J:LIN=0
7620 IF LEN(P$)>LL THEN L=LL:GOTO 7550
7630 IF CR AND LEN(P$)>0 THEN 7530
7640 RETURN
```

Subroutine to re-adjust lines in memory so that they fit properly on the screen after editing.

```
8000 ? CL$:POKE 752,1:POSITION 2,5:? "
Re-justifying text..."
8010 TL=LEN(T$):N=EL+NL-1:C=LP(N)+1:SL
=C-1:CH=C-LP(N-1)
8020 C$=T$(C,C)
8030 IF C$=CR$ THEN 8100
8035 IF C=TL THEN FL=1:SLOC=SL-LP(N-1)
:GOTO 8100
8040 IF C$=" " THEN SL=C
8050 IF CH<LWID-1 THEN CH=CH+1:C=C+1:G
OTO 8020
8060 IF SL=LP(N-1) THEN SL=C
8070 LP(N)=SL:C=SL+1:N=N+1:CH=1:GOTO 8
020
8100 LP(N)=C
8110 IF LP(N)=LP(N+1) THEN LN=LN-1:FOR
 I=N TO LN-1:LP(I)=LP(I+1):NEXT I
8120 RETURN
```

Subroutine to edit lines of text.

```
9000 FL=0:IF CHAR>1 THEN LP(LN)=LEN(T$
):LN=LN+1:LP(LN)=LEN(T$):L$="":FL=1
9005 POKE 752,1:IT=I:IF I>21 THEN V1=2
2:GOTO 9040
9010 V1=I+1:POSITION 2,VV
9020 X=21:IF X>LN-1 THEN X=LN-1
9025 IF X=IT THEN 9040
9030 FOR I=IT+1 TO X:? T$(LP(I-1)+1,LP
(I)):NEXT I
9040 EL=V1+(IT>21)*(IT-21)-1
9050 POSITION 2,0:Q$="UP/DN:Move RTN:E
dit D/X:Del ESC:Exit ":? Q$:? LIN$
9080 C=ASC(T$(LP(EL-1)+1)):POSITION 2,
V1:PRINT CHR$(C+128);:GET #1,X:IF X<>1
55 THEN X=X-128*(X>127):X=X-32*(X>90)
9085 POSITION 2,V1:? CHR$(C);:IF X=6 A
ND ED=0 THEN 11000
9090 IF X<>28 THEN 9130
9100 IF V1>2 THEN V1=V1-1:EL=EL-1:GOTO
 9080
9110 IF EL=1 THEN 9080
9115 EL=EL-5:NN=4:IF EL<1 THEN NN=EL+3
:EL=1
9120 FOR I=EL+NN TO EL STEP -1:POSITIO
N 2,2:? CHR$(157);T$(LP(I-1)+1,LP(I)):
NEXT I:POSITION 2,23:? S$;:GOTO 9080
9130 IF X<>29 THEN 9180
9140 IF EL>=LN-1-FL THEN 9080
9150 EL=EL+1
9160 IF V1<22 THEN V1=V1+1:GOTO 9080
9165 NN=4:IF NN>LN-EL-1-FL THEN NN=LN-
EL-1-FL
9170 EL=EL+NN:POSITION 2,23:FOR I=EL-N
N TO EL:? T$(LP(I-1)+1,LP(I)):NEXT I:G
OTO 9050
9180 IF X=27 THEN 9580
9190 IF X<>4 OR V1=2 THEN 9250
9200 NC=LP(EL)-LP(EL-1):IF EL=LN-1 THE
N T$=T$(1,LP(EL-1)):GOTO 9205
9202 T$(LP(EL-1)+1)=T$(LP(EL)+1)
9205 FOR J=EL TO LN-1:LP(J)=LP(J+1)-NC
:NEXT J
9210 X=22-V1:IF X>LN-EL-2 THEN X=-1
9220 POSITION 2,V1:? CHR$(156);
9225 IF X>-1 THEN POSITION 2,22:? S$:?
 CHR$(28);:IF EL<LN-1 THEN ? T$(LP(EL+
X-1)+1,LP(EL+X))
9230 IF EL=LN-1-FL THEN V1=V1-1:EL=EL-
1
9240 LN=LN-1:GOTO 9080
9250 IF X<>24 THEN 9310
```

```
9260 POSITION 2,0:? "Delete from here
to the end of text?";:GET #1,X:X=X-128
*(X>127):X=X-32*(X>90)
9265 IF X<>89 THEN 9060
9270 LN=EL:L$="":CHAR=1:SLOC=0:IF LN>1
 THEN T$=T$(1,LP(LN-1)):GOTO 9280
9275 T$=""
9280 TL=LEN(T$):GOTO 9580
9310 IF X<>155 THEN 9080
9320 L1=EL-8:IF L1<1 THEN L1=1
9330 L2=L1+16:IF L2>LN-1 THEN L2=LN-1
9340 Q$="Type new line below (^F to fi
nish)  ":? CL$;Q$:? LIN$
9350 FOR J=L1 TO EL:? T$(LP(J-1)+1,LP(
J)):NEXT J
9360 ? :? :? :?
9370 IF L2>EL THEN FOR J=EL+1 TO L2:?
T$(LP(J-1)+1,LP(J)):NEXT J
9380 POKE 752,0:POSITION 2,EL-L1+2
9390 TLN=LN
9410 C1=CHAR:S1=SLOC:CHAR=1:SLOC=0
9420 ED=1:GOSUB 500
9430 CHAR=C1:SLOC=S1
9450 NL=LN-TLN:NC=LP(LN-1)-LP(TLN-1)
9480 IF EL=LN-1 THEN T$=T$(1.LP(EL-1))
:GOTO 9490
9485 T$(LP(EL-1)+1)=T$(LP(EL)+1)
9490 CC=LP(EL)-LP(EL-1):FOR J=EL TO LN
-1:LP(J)=LP(J+1)-CC:NEXT J:IF NL=0 THE
N LN=TLN-1:GOTO 9580
9500 X$=T$(LEN(T$)):IF X$<>CR$ AND X$<
>" " THEN T$(LEN(T$)+1)=" ":NC=NC+1:LP
(LN-2)=LP(LN-2)+1:LP(LN-1)=LP(LN-1)+1
9502 CN=LP(EL-1):FOR I=LEN(T$) TO CN+1
 STEP -NC:IF I<NC THEN T$(CN+NC+1,I+NC
)=T$(CN+1,I):GOTO 9504
9503 T$(I+1,I+NC)=T$(I-NC+1,I)
9504 NEXT I
9505 T$(CN+1,CN+NC)=T$(LEN(T$)-NC+1)
9506 FOR I=LN-2 TO EL STEP -1:LP(I+NL)
=LP(I)+NC:NEXT I:J=TLN+NL-1
9508 K=LP(LN-2)-LP(EL-1):FOR I=0 TO NL
-1:LP(EL+I)=LP(J+I)-K:NEXT I
9510 T$=T$(1,LEN(T$)-NC)
9530 LN=TLN+NL-1:IF X$=CR$ OR EL=LN-NL
 THEN 9580
9550 SS=1:P$=T$(LP(EL+NL-1)+1,LP(EL+NL
)):LL=LEN(P$)
9560 IF P$(SS,SS)<>" " AND SS<LL THEN
SS=SS+1:GOTO 9560
9570 IF LP(EL+NL-1)-LP(EL+NL-2)+SS<=LW
ID THEN GOSUB 8000
9580 TL=LEN(T$):IF FL THEN LN=LN-1:L$=
T$(LP(LN-1)+1,LP(LN)):CHAR=LEN(L$)+1:T
L=LP(LN-1)
9600 STP=0:RETURN
```

Subroutine to convert lower case to upper case.

```
10000 FOR AZ=1 TO LEN(F$):F$(AZ,AZ)=CH
R$(ASC(F$(AZ,AZ))-32*F$(AZ,AZ)>"Z"):NE
XT AZ:RETURN
```

Subroutine to perform the find function.

```
11000 PLOT 0,0:DRAWTO 39,0:POSITION 2,
0:? "Search string";:INPUT AZ$:X=EL-1:
IF AZ$="" THEN 9050
11010 POSITION 2,0:? "Searching for '"
;AZ$;"'"
11020 X=X+1:IF X=LN THEN POSITION 2,0:
? CHR$(253);"String not found, press a
ny key":GET #1,AZ:GOTO 9050
11030 L$=T$(LP(X-1)+1,LP(X)):IF LEN(L$
)-LEN(AZ$)<0 THEN 11010
11040 FOR AZ=1 TO LEN(L$)-LEN(AZ$)+1:I
F L$(AZ,AZ+LEN(AZ$)-1)<>AZ$ THEN NEXT
AZ:GOTO 11020
11050 EL=X:L$="":GOTO 9320
```

Error-handling routine.

```
12000 FOR AZ=1 TO LN-1:FOR X=LP(AZ-1)+
1 TO LP(AZ)
20000 CLOSE #2:E=PEEK(195)
20010 PLOT 0,0:DRAWTO 39,0
20020 PLOT 0,1:DRAWTO 39,1
20030 POSITION 2,0:? "Error: Code ";E;
"; press any key":? LIN$
20040 GET #1,X:TRAP 20000
20050 IF PF=1 THEN PF=0:GOSUB 3000
20060 ED=0:GOTO 200
```

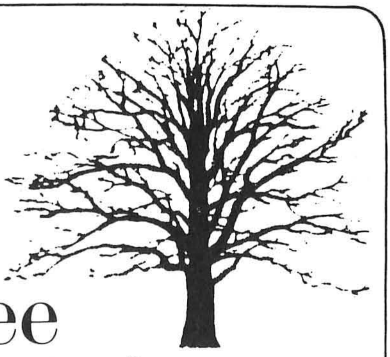
SWAT TABLE

For **ATARI® MICROTEXT 2.0**

LINES	SWAT CODE	LENGTH
10 - 125	TK	540
130 - 510	MV	576
520 - 760	PE	286
780 - 1050	BB	324
1060 - 2060	RY	286
2070 - 2950	FX	472
3000 - 3135	LH	391
3140 - 4020	RY	457
4030 - 4200	RA	571
4205 - 5075	RP	458
5080 - 5205	HL	503
5210 - 6000	RU	276
6080 - 7080	ER	534
7090 - 7530	XU	270
7540 - 8000	UZ	408
8010 - 9000	AK	498
9005 - 9085	PB	510
9090 - 9170	PF	526
9180 - 9260	VV	624
9265 - 9380	XM	474
9390 - 9502	TK	561
9503 - 9580	OG	542
9600 - 11050	PL	514
12000 - 20060	MC	312

Microtext Microte
Microtext Microtext Microt
.0 Microtext 2.0
text 2.0 Microtext 2.0 Mic
icrotext 2.0 Micr
2.0 Microtext 2

DV BONUS



Family Tree Organizer

by Fred Coffey

Family Tree Organizer is a genealogical data management package for the Atari® 400/800/1200 with Atari BASIC, 32K RAM, a disk drive and a printer.
At present the programs are written for an Epson printer, but with the aid of the special section on Printer Control Codes, adapting it for other printers should not be difficult, as long as they have "compressed character" mode.

This data management package organizes and presents genealogical data. It creates data files on your relatives, including unlimited biographical information. You can retrieve data in a variety of formats (individual biography, pedigree charts, and descendant tabulation). After you edit or update any files, the computer automatically traces any new or redefined relationships.

After booting DOS, type "RUN D:Family" to see the *Family Tree Organizer's* main menu:

- FILEPROG — Creates/edits disk data files on your relatives, and prints out a nicely formatted biography for your records (Figure 1)

- DIRPROG — Maintains and prints a data file directory, (Figure 2)

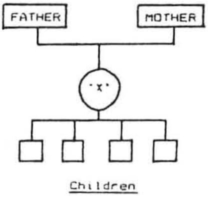
- CHARPROG — Prints a pedigree chart for you (or any of your relatives) by tracing the defined ancestor relationships (Figure 3)

- DESLIST — Provides a listing of all the descendants it can find for any given person (Figure 4)

Planting the Family Tree

Make a copy of the Master Disk — it will get a lot of hard use. Next, format a blank disk to receive the files you will create. Actually you can use the Master Disk to hold data if you prefer, but less space is available. Label your data disk "Disk Number 1." If and when it's full you can add more disks. Each can hold 64 files, representing several hundred relatives. The program can keep track of about 500 files (i.e. eight to ten disks and two thousand relatives).

You can start anywhere — starting with yourself works well. Just choose a Person X and prepare to enter the information. Specifically, the computer is interested in X's parents (if known) and children (if any) for family tracing purposes, and will accept an unlimited amount of biographical information. It creates a file on X defining the following relationships:



But didn't we forget that X might also have a spouse? Actually, he/she might have several (not necessarily at the same time) and that's the problem. So we'll use the children to establish the marriage connection(s), so when you tell the computer about the children you will also tell it who the other parent is. To show a marriage with no children, we lie a bit, and tell the computer they had a child called "NO CHILDREN." The computer will be quite satisfied.

Next, we create similar files for X's father, his mother, their parents, and so on through the generations. When we ask the computer for a pedigree chart on X, it searches the directory for files on each person it encounters, and traces up or down the tree as appropriate until it finds no more information. The limit is fifteen generations when tracing descendants.

File Generation

To create a file, insert the Master Disk, and type RUN "D:FILEPROG". After the disk stops, take it out and insert the "Disk Number 1" you previously formatted. The screen prompts for a file name, or you could hit RETURN to start a new file. We'll discuss the file name for retrieving an old file later, so just hit RETURN to indicate a new file.

Now just follow the prompts. Remember the chart we showed earlier for X? The computer starts by asking for X's given names, family name and birth year. You must use only capital letters for names. If you don't know the exact birth year, the program cheerfully accepts dates like "192x."

The computer now draws a small box and asks for biographical information. The box has room for five lines of 32 characters. You can enter any potentially useful facts, in any format — dates and places of birth, marriage, death and burial might be informative. Avoid Atari graphics (control-shift) characters. Figure 3 shows one user's choice for the type of information to provide here.

The computer moves on to collect data on X's parents. Just answer the prompts, remembering to use only capital letters for names.

For the section on children, the computer displays sample data in the desired format, then waits for instructions. To edit the screen, type "E" (and RETURN) then enter data as required. Type "S" to save the data and move on to the next child. The screen retains information that might be used again, so you don't have to type it twice. When you finish entering data for all the children, type "F." to finish.

The final section is for an extended biography or other notes. You enter text a "line" at a time, with the last several lines displayed on the screen. The word "line" refers to a 75 character printer line, which in fact takes up two screen lines. Type until the cursor nears the end of the second line, then hit RETURN. To put an extra blank line between paragraphs, just hit RETURN without entering anything. When finished, enter "@" to exit this portion.

DV BONUS

DV BONUS

The computer now asks if you want to save this data in file "llllyyyy.nnn". After a bit of reflection, you'll discover that "llll" represents the first four characters of X's last name, "yyyy" is year of birth, and "nnn" the first three characters of the first name. You've just decoded the secret of the computer's filing system. Write the file name on a piece of paper — you'll need it again shortly. If you want to save the file, type "YES" and press RETURN. Spell out the word "YES" — it won't accept anything less because I deliberately made it difficult to overwrite an important file accidentally. The computer asks you to make sure the correct disk is inserted. Press RETURN and the file is saved. Don't rush on to create a file on your next relative. Review the editing functions below and save yourself some typing.

Editing Children

Type RUN again. This time, when prompted, enter the name of the file previously created. If you can't remember the name, use the DOS command to see the disk directory. We'll put it into a different kind of directory later. The program asks if you want to update the file, or print it. Choose "update." The program retrieves data from the disk, waiting after each step to see if you want to change anything. Change whatever you want, then hit RETURN when it's right.

When you reach the files on children you have several choices:

COMMAND	PURPOSE
S	Save
D	Delete
E	Edit
I	Insert another child's file before this one
F	Finish the child data entry
X	Special (we'll explain that later)

At the biography section, the program retrieves each line from disk and waits. You can edit (just re-type the line, or move the cursor to where corrections are needed and overstrike), delete (*), insert (+), or quit (@). When done, save the file as before.

Typing Shortcuts

The most important shortcut when creating a new file is to recycle one you've already made on someone else with a lot in common. A spouse will have the same children, a brother the same parents, etc. Instead of indicating a "new file," ask for the file of the "similar person" and just edit it as necessary.

That "special" option mentioned earlier for editing the children's files will create the other parent's file from the spouse's, since some or all of the children are identical. When the first child comes up for editing, do a normal edit to include his other parent, then use special option "X". The computer reads this command as "save the data on the screen, then bring up the next child without changing the parent data on the screen."

The Directory

Create more files now if you want, but note the name of each file as you go. Before we can do much else, we must record these file names in the "directory." The chart printing programs use this directory to determine if various files exist and where they are. When you're ready to make entries in the directory, insert the Master Disk and type RUN "D:DIRPROG". The program asks for names of files you want saved or deleted. Type in each name, then when asked, indicate the disk number for that file. It continues to ask for names until you finish by pressing RETURN without an entry. Answer Yes to the prompt if you want to print the directory. Note that the directory file resides on the Master Disk, because several programs use it when they are executed, and the directory includes individuals on all your data disks.

Pedigree Chart

Insert the Master Disk and type RUN "D:CHARPROG". The program asks for the name of the initial file — the person whose progenitors you want to find. Next, it asks if you have a standard MX-80 printer. If not, or if you have some trouble with the printout, you should read the special section on Printer Control Codes to find some solutions. It now announces it's looking for the first file and asks you to insert the appropriate disk. If it can't find all the required files on that disk, it may ask you to change disks one or more times before starting to print. If you haven't put all the file names into the directory, it won't be able to look for them.

Printing the Family Tree

Load the Master Disk, and type RUN "D:DESLIST". The program asks you to enter a "title" to be printed at the top of each page. Next enter the file name for the person whose descendants you want tabulated. If necessary, the program occasionally will ask you to insert another disk. Again, keeping the directory up to date is important.

After tracing all known descendants of the person selected, it asks for another file name. If you have all the information you want, type DONE to return to the main menu. However, if you want to continue the tabulation starting with a different ancestor, enter the file name, and the tabulation continues from there. If it finds it is about to repeat a listing, it inserts a page reference and skips ahead.

A good approach is to generate a pedigree chart (or charts, if it goes back more than five generations) for a given person. Then tabulate (in a single run) the descendants for all the most distant ancestors. When finished you will have a tabulation of all that person's known relatives.

Large Families

When using more than one disk to hold all the records, try to keep logical family groups together. Avoid entering names randomly. For example, put your father's family on one disk and your mother's on another. Keep in mind what sort of charts you will be making and for whom. If you are keeping track of cousins who will appear only on descendant lists, you might pack two or more generations into one file.

To see how to do this, consider the descendant list. It indicates various generations by indenting an additional two spaces for each generation. Simply put both the children and grandchildren into a person's data file, but insert two spaces in front of the first name of each grandchild. Then on the printout it will show up as if you had created a file on an extra generation.

DV BONUS

FAMILY TREE ORGANIZER

DV BONUS

Printer Control Codes

These programs were written for an Epson MX-80, using its graphics characters (identical to the TRS-80 set), and probably will not work on other printers. In fact it may not work as intended on all Epsons — later models don't have the graphics font.

The principal difficulty will be drawing the pedigree charts, and a "quick fix" is available as an option in executing that program (CHARPROG). When you run the program it asks if you have a standard MX-80; if you reply "no" it avoids using the graphics characters. If problems persist, or if you are not satisfied with the appearance of your charts, read on.

First, determine whether your printer uses the following control character codes. If not, you will have to make substantial changes by locating all the codes and changing them.

ASCII CONTROL CODE	PURPOSE
CHR$(27);CHR$(48)	Switch to 1/8 inch line spacing
CHR$(12 + 128)	Advance paper to top of form
CHR$(14)	Turn on double-width printing
CHR$(15)	Turn on compressed character mode (132 characters per line)
CHR$(20)	Turn off double width.

If these control codes are compatible or translatable, you still may need to find the corresponding graphics characters (and their ASCII codes) on your printer to draw the special sections on the pedigree chart. The MX-80 characters are identified below:

CHARACTER #	SHAPE	CHARACTER #	SHAPE
165		188	
170		189	
172		202	
173		204	
174		206	
181		43	+

To facilitate adapting the programs to other printers, I have located the printer control codes for each program in the 20000 series of line numbers, and have included numerous REM statements at that location. These graphics characters apply only to the "CHARPROG" program. You can change them to characters suitable for your printer by changing the code number on the right side of each "CHARNUM =" statement in lines 21210 through 21260.

The "CHARPROG" program also uses eight lines per inch for printing, versus the standard six lines per inch. If your printer cannot do this, simply delete line 20200. Unfortunately your pedigree charts will be somewhat longer than an eleven inch sheet of paper. You could add a line of code to count the lines printed and skip to the next page if this bothers you.

The programs also use double-strike and double-width print modes. If your printer lacks this ability, override the option as specified in the "REM" statements. Your charts will look a bit different from the examples shown, but should be quite readable.

SCREAMIN' DEMON

by Greg Schroeder

Screamin' Demon is a game for the Atari® computer with 16K of RAM (24K with disk) and one joystick.

Screamin' Demon is an arcade style pinball game complete with multiple flippers, bumpers, flashing lights and sound effects that add an eerie flavor to the game. For even more excitement, the ball may be played on not one, but three levels, if you can find the way to get to them.

Start the game by pulling back on the joystick. This simulates the plunger action in real pinball. Releasing the stick sends the ball on its way. The bumpers on all levels are activated by pressing the joystick button.

Hitting the ball into one of the various secret traps on the screen either sends the ball to another level, or scores bonus points. You'll earn an extra ball for every 50,000 points.

Variables

BALL: Number of balls left.
BONUS: Score at which bonus credits are achieved.
CR: Number of bonus credits.
D: Start of display list.
DISK: Disk flag for high score table (1 for disk, 0 for cassette).
H(x): High scores (all time highs if disk system).
HS: Most recent high score.
H$: High score initials.
I,J,J1,J2,J3,K: Miscellaneous variables.
L: Temporary ball level.
LEV: Current playing level the ball is on.
N: Number of initials typed in.
NT: Current note counter.
S: Game starting flag, stick status.
SC: Score.
T: Length of time flippers have been pressed.
TI: Timing variable for background music.
V: Volume of explosion.
X,Y,X1,Y1: Position of ball on screen.
XM,YM: Motion variables for X,Y movement of ball.
X$: Machine language code for flippers.
Z,ZZ,Z1,Z3: Miscellaneous screen contents variables.

SCREAMIN' DEMON

```
SS SS SS SS SS SS SS SS SS SS SS
SS                            SS
SS          Atari BASIC       SS
SS      'Screamin' Demon'     SS
SS   Author: Greg Schroeder   SS
SS      Copyright © 1983      SS
SS SoftSide Publications, Inc SS
SS                            SS
SS SS SS SS SS SS SS SS SS SS SS
```

If you don't wish to type this program, it is available on issue #42 SoftSide CV and DV.

Initialization
Data for machine language program stored in X$ that puts on and erases all ten flippers on the screen.

```
1 DATA 104,162,14,189,50,6,24,101,89,1
33,205,189,36,6,24,101,88,133,204,144,
2
2 DATA 230,205,160,0,177,204,93,64,6,1
45,204,202,208,224,96,0,224,223,2,1,0,
252,251,139,136,135,216,211,210,0
3 DATA 2,2,3,3,3,2,2,2,2,2,1,1,1,0,42,
168,170,160,10,42,168,170,160,10,170,1
60,10
```

GOSUB to initialize X$, then do high-score initialization. If DISK=1 then the program will load the high-score table from the disk, otherwise, the high scores change whenever the program is rerun. Readers without disk should make DISK=0.

```
4 GOSUB 870:DISK=1
5 TRAP 10:DIM A(2),H(10),H$(30):H$="
                              ":IF DISK
=0 THEN FOR I=1 TO 10:H(I)=0:NEXT I:GO
TO 10
6 TRAP 8:OPEN #1,4,0,"D:HISCORES":FOR
I=1 TO 10:INPUT #1,H:H(I)=H:NEXT I:INP
UT #1;H$:CLOSE #1:GOTO 10
8 CLOSE #1:OPEN #1,8,0,"D:HISCORES":FO
R I=1 TO 10:? #1,0:NEXT I:? #1,H$:CLOS
E #1:GOTO 6
10 GRAPHICS 5:GOSUB 690:GOSUB 700:REST
ORE 90
25 LEV=1:BALL=3:SC=0:CR=0:BONUS=50000:
SOFF=0:GOSUB 500:IF S=1 THEN S=0:GOTO
55
```

Plunger routine that starts off each new ball.

```
55 TRAP 250:X=53:Y=26:XM=0:YM=0:COLOR
2:PLOT X,Y
60 IF STICK(0)=13 THEN COLOR 0:PLOT X,
Y:Y=Y+(Y<38):COLOR 2:PLOT X,Y:SOUND 0,
Y*3,10,10:FOR J=1 TO 30:NEXT J:GOTO 60
65 IF Y=26 THEN 60
70 SOUND 0,0,0,0:YM=-((Y-26)/24+0.5)
75 COLOR 0:PLOT X,Y:Y=Y+YM:COLOR 2:PLO
T X,Y:FOR J=1 TO 20:NEXT J:IF Y<3 THEN
 COLOR 0:PLOT X,Y:GOTO 80
76 GOTO 75
80 Y=1:X=52:XM=YM:YM=RND(0)/4:FOR I=1
TO 25:COLOR 0:PLOT X,Y:X=X+XM:COLOR 2:
PLOT X,Y:FOR J=1 TO 20:NEXT J:NEXT I
85 COLOR 0:PLOT X,Y:COLOR 1:PLOT 39,1:
PLOT 39,2:X1=INT(X):Y1=INT(Y):POKE 77,
0
```

Data for background music.

```
90 DATA 60,2,64,2,60,2,81,2,96,1,81,2,
121,4,60,2,64,2,60,2,81,2,96,1,81,2,12
1,4,60,2,53,2,50,2,53,1,50,2,60,4
92 DATA 53,2,57,1,53,2,64,2,60,2,81,2,
96,1,81,2,121,4,60,2,64,2,60,2,81,2,96
,1,81,2,121,4,60,2,64,2,60,2,81,2,96,1
94 DATA 81,2,121,4,60,2,53,2,50,2,53,1
,50,2,60,4,60,2,53,2,57,1,53,2,64,2,60
,2,64,2,60,1,64,2,60,4,60,2,53,2,47,2,
53,2
96 DATA 47,2,60,2,53,2,57,1,53,2,64,2,
60,2,81,2,96,1,81,2,121,4
98 DATA 182,4,193,4,243,2,217,8
100 IF V>1 THEN V=V-1:SOUND 0,100,8,V:
GOTO 102
```

Main program loop.

```
101 SOUND 0,0,0,0
102 COLOR 0:PLOT X1,Y1:YM=YM+0.015-0.0
15*(YM=1):Z=0:Z1=0:IF XM<>0 AND YM>0 T
HEN XM=XM+0.01-0.02*(XM>0)
103 IF PEEK(764)=62 THEN POKE 764,255:
SOFF=1-SOFF:SOUND 1,0,0,0
```

Play background music.

```
104 IF SOFF THEN 110
105 IF TI>0 THEN TI=TI-1:GOTO 110
106 SOUND 1,0,0,0:NT=NT+1:IF NT=75 THE
N NT=1:RESTORE 90
```

```
108 READ N,TI:SOUND 1,N,10,4:TI=TI
110 LOCATE INT(X+XM),INT(Y+YM),Z:IF Z=
0 THEN X=X+XM:Y=Y+YM:X1=INT(X):Y1=INT(
Y):COLOR 2:PLOT X1,Y1:GOTO 150
111 GOTO 115
```

Make sounds, increase score, and move the ball in the direction according to the type of bumper hit.

```
115 SOUND 0,200-Z*50,12,10:SC=SC+(Z-1)
*50*LEV:Z3=PEEK(707+Z):ZZ=Z:POKE 707+Z
,Z3+6:POKE 656,0:POKE 657,30:? SC;
116 LOCATE INT(X+XM),Y1,Z:LOCATE X1,IN
T(Y+YM),Z1:POKE 707+ZZ,Z3
120 IF Z<>0 AND Z1=0 THEN XM=-XM:GOTO
150
130 IF Z=0 AND Z1<>0 THEN YM=-YM:GOTO
137
135 XM=-XM:YM=-YM
137 IF XM>-0.2 AND XM<0.2 THEN XM=(2*R
ND(0)-1)*YM
139 GOTO 150
150 IF Z=1 OR Z1=1 THEN GOSUB 300
152 IF (Z=2 OR Z1=2) AND T>0 THEN GOSU
B 175+5*LEV
154 IF Z=3 OR Z1=3 THEN YM=SGN(YM):V=1
5
156 IF STRIG(0)=0 THEN 200
158 IF T>0 THEN T=0:COLOR 0:PLOT X1,Y1
:A=USR(1536):COLOR 2:PLOT X1,Y1
160 GOTO 100
180 J=22:GOSUB 195:J=31:GOSUB 195:J=37
:GOSUB 195:GOTO 199
185 J=37:GOSUB 195:GOTO 199
190 J=35:GOSUB 195:GOTO 199
195 IF Y1=J THEN 197
196 RETURN
197 IF T>5 THEN YM=YM/3:RETURN
198 YM=-0.5-(6-T)/10:RETURN
199 RETURN
```

Flipper control.

```
200 IF T>0 THEN T=T+1:GOTO 100
210 T=1:COLOR 0:PLOT X1,Y1:SOUND 0,200
,10,10:A=USR(1536)
215 LOCATE X1,Y1,Z:IF Z=0 THEN 220
217 Y=Y-1:Y1=Y1-1:GOTO 115
220 COLOR 2:PLOT X1,Y1:GOTO 100
```

Ball went to bottom of the screen. If the ball was on level one, you lose the ball and it checks for bonus credits. If the ball was on the second or third levels, it returns it to level one and adds bonus score.

```
250 IF Y<1 THEN Y=1:Y1=INT(Y)
255 IF Y>38 THEN SOUND 1,0,0,0:GOTO 27
5
270 TRAP 250:GOTO PEEK(187)*256+PEEK(1
86)
275 IF LEV<>1 THEN L=1:GOSUB 340:GOTO
270
277 FOR I=1 TO 50:NEXT I
280 RESTORE 98:FOR I=1 TO 4:READ N,TI:
SOUND 1,N,10,10:SOUND 2,200,12,10:FOR
J=1 TO TI*65:NEXT J:NEXT I:RESTORE 90
282 IF SC>=BONUS THEN 295
285 NT=0:BALL=BALL-1:POKE 656,0:POKE 6
57,11:? CHR$(176+BALL);:IF BALL=0 THEN
 600
290 SOUND 1,0,0,0:SOUND 2,0,0,0:FOR I=
1 TO 500:NEXT I:GOTO 55
295 CR=CR+1:BONUS=BONUS+50000:POKE 656
,2:POKE 657,11:? CHR$(176+CR);
297 FOR I=1 TO 4:SOUND 1,100,10,15:FOR
 J=1 TO 20:NEXT J:SOUND 1,0,0,0:FOR J=
1 TO 20:NEXT J:NEXT I:GOTO 285
```

Check to see if the ball went in a trap door or bonus slot and change the ball level, add to score, and make sound effects.

```
300 SC=SC+10:GOTO 300+5*LEV
305 J=44:K=12:L=INT(RND(0)*2)+2:GOSUB
335:J=45:GOSUB 335:J=39:K=14:L=4:GOSUB
 335:J=41:L=2:GOSUB 335
307 J=43:L=3:GOSUB 335:J=45:L=4:GOSUB
335:GOTO 390
310 J=7:K=33:L=1:GOSUB 335:J=24:GOSUB
335:GOTO 390
315 J=55:K=23:L=1:GOSUB 335:J=72:GOSUB
 335:GOTO 390
335 IF X1=J AND Y1=K THEN POP :GOTO 33
5+5*L
337 RETURN
340 SOUND 1,0,0,0:LEV=L:SC=SC+2000:X=I
NT(RND(0)*16)+33:Y=19
```

```
342 FOR K=-80 TO 90 STEP 10:FOR J=0 TO
 16 STEP 2:SOUND 0,100-ABS(K)+J*SGN(K)
,8,12:NEXT J:NEXT K:SOUND 0,0,0,0:GOTO
 360
345 SOUND 1,0,0,0:LEV=L:SC=SC+4000:X=I
NT(RND(0)*12)+10:Y=28
346 FOR K=1 TO 4:J1=INT(RND(0)*15)*5+1
50:J2=INT(RND(0)*5)*5+100:FOR J=J2 TO
J1 STEP 2
348 SOUND 0,J,8,12:J3=14-J3:POKE 712,J
3:NEXT J:NEXT K:POKE 712,0:SOUND 0,0,0
,0:GOTO 360
350 SOUND 1,0,0,0:LEV=L:SC=SC+4000:X=I
NT(RND(0)*12)+58:Y=29:GOTO 346
355 FOR K=1 TO 3:FOR J=255 TO 0 STEP -
5:SOUND 0,J,4,10:SOUND 1,255-J,8,10:SC
=SC+25:POKE 656,0:POKE 657,30:? SC;
357 NEXT J:NEXT K:SOUND 0,0,0,0:SOUND
1,0,0,0:GOTO 360
360 X1=INT(X):Y1=INT(Y):YM=1
390 POKE 656,0:POKE 657,30:? SC;:RETUR
N
```

High score routine.

```
400 GRAPHICS 1:GOSUB 690:POSITION 5,2:
? #6;"aaaarrgh":POSITION 0,4:? #6;"YOU
 HAVE UPSET THE  SPIRITS BY MAKING"
405 ? #6;"TOO MANY POINTS.    NOW YOU
MUST ENTER  YOUR INITIALS IN OURPERMAN
ENT RECORDS"
410 ? #6;"SO WE MAY REFERENCE THEM WHE
N YOUR TIME COMES..........."
420 ? " CHOOSE THE LETTER WITH THE JOY
STICK":? "       PRESS THE BUTTON TO EN
TER"
425 POSITION 3,15:? #6;"abcdefghijklm
      nopqrstuvwxyz"
430 X=3:Y=15:L=1:N=0
435 SOUND 0,0,0,0:S=STICK(0):IF S=15 A
ND STRIG(0)=1 THEN 435
440 IF STRIG(0)=0 THEN A(N)=L+64:N=N+1
:SOUND 0,100,12,10:FOR I=1 TO 50:NEXT
I:GOTO 490
450 POSITION X,Y:? #6;CHR$(L+224);:IF
S=7 THEN L=L+1-26*(L=26)
455 IF S=11 THEN L=L-1+26*(L=1)
460 Y=15:IF L>13 THEN Y=16
465 X=L-INT(L/14)*13+2:POSITION X,Y:?
#6;CHR$(L+64);:SOUND 0,RND(0)*150+50,1
0,10:FOR I=1 TO 5:NEXT I:SOUND 0,0,0,0
466 GOTO 435
470 SOUND 0,0,0,0:FOR I=1 TO 10:IF HS<
=H(I) THEN NEXT I:GOTO 495
475 IF I=10 THEN H(I)=HS:GOSUB 485:GOT
O 490
480 FOR J=10 TO I+1 STEP -1:H(J)=H(J-1
):H$(J*3-2,J*3)=H$((J-1)*3-2,(J-1)*3):
NEXT J:H(I)=HS:GOSUB 485:GOTO 495
485 H$(I*3-2,I*3-2)=CHR$(A(0)):H$(I*3-
1,I*3-1)=CHR$(A(1)):H$(I*3,I*3)=CHR$(A
(2)):RETURN
490 POSITION 7+N,18:? #6;CHR$(A(N-1)+3
2);:IF N=3 THEN 470
492 GOTO 435
495 IF DISK=0 THEN RETURN
497 OPEN #1,8,0,"D:HISCORES":FOR I=1 T
O 10:? #1,H(I):NEXT I:? #1;H$:CLOSE #1
:RETURN
```

Screen display routine.

```
500 POKE 559,0:? "}":? #6;"}"
502 COLOR 1:PLOT 6,39:DRAWTO 6,22:DRAW
TO 25,22:DRAWTO 25,39:DRAWTO 25,0:DRAW
TO 54,0:DRAWTO 54,39:DRAWTO 54,22:DRAW
TO 73,22
505 DRAWTO 73,39:PLOT 52,39:DRAWTO 52,
3:DRAWTO 37,3:DRAWTO 37,10:PLOT 37,13:
DRAWTO 47,13
510 PLOT 28,30:DRAWTO 28,39:PLOT 49,30
:DRAWTO 49,39:PLOT 55,34
515 DRAWTO 55,39:PLOT 72,34:DRAWTO 72,
39
520 COLOR 2:PLOT 55,28:DRAWTO 55,33:PL
OT 72,28:DRAWTO 72,33:COLOR 1
525 FOR I=38 TO 46 STEP 2:PLOT I,14:NE
XT I:COLOR 3:PLOT 40,4:DRAWTO 49,4:PLO
T 28,6:PLOT 28,7
530 PLOT 34,6:PLOT 34,7:COLOR 2:PLOT 3
1,6:PLOT 31,7:PLOT 42,7:PLOT 43,7:PLOT
 46,7:PLOT 47,7
535 COLOR 3:PLOT 38,6:DRAWTO 38,10:PLO
T 51,6:DRAWTO 51,10
540 COLOR 1:PLOT 71,23:PLOT 56,23:PLOT
 7,32:PLOT 24,32
545 I=28:FOR J=32 TO 36 STEP 2:FOR K=0
 TO 1:PLOT I+K,J:DRAWTO I+K,39:NEXT K:
I=I+2:NEXT J
550 I=49:FOR J=32 TO 36 STEP 2:FOR K=0
 TO 1:PLOT I-K,J:DRAWTO I-K,39:NEXT K:
I=I-2:NEXT J
```

```
555 FOR I=58 TO 68 STEP 10:FOR J=0 TO
1:PLOT I+J,34:DRAWTO I+J,39:NEXT J:NEX
T I
560 FOR I=7 TO 20 STEP 13:FOR J=0 TO 4
:PLOT I+J,34:DRAWTO I+J,39:NEXT J:NEXT
 I
564 COLOR 2:PLOT 25,23:PLOT 52,23:PLOT
 29,32:PLOT 48,32:PLOT 33,38:PLOT 44,3
8
566 PLOT 11,38:PLOT 20,38:PLOT 59,36:P
LOT 68,36
570 COLOR 2:J=13:K=23:GOSUB 580:J=17:G
OSUB 580:J=28:K=16:GOSUB 580:J=32:GOSU
B 580:J=38:K=4:GOSUB 580:J=50:GOSUB 58
0
572 J=63:K=23:GOSUB 580:J=61:K=38:GOSU
B 580:J=65:K=38:GOSUB 580:COLOR 3
575 J=9:K=26:GOSUB 580:J=10:K=30:GOSUB
 580:J=20:GOSUB 580:J=21:K=26:GOSUB 58
0:J=30:GOSUB 580:J=46:GOSUB 580
576 J=42:K=11:GOSUB 580:J=46:GOSUB 580
577 J=58:K=32:GOSUB 580:J=68:GOSUB 580
:J=60:K=26:GOSUB 580:J=66:GOSUB 580
578 GOTO 590
580 PLOT J,K:PLOT J+1,K:PLOT J+1,K+1:P
LOT J,K+1:RETURN
590 ? "}":FOR I=0 TO 3:POKE 656,I:POKE
 657,0:? "____________________";:NEXT
I:POKE 656,0:POKE 657,2
592 ? "BALLS  - ";CHR$(176+BALL);:POKE
 656,2:POKE 657,2:? "CREDIT - ";CHR$(1
76+CR);
595 POKE 656,0:POKE 657,21:? "SCORE -
 ";SC;:POKE 656,2:POKE 657,21:? "HIGH
 -  ";H(1);:POKE 559,34:RETURN
```

Game over. Check for bonus credits and continue game if there are some, otherwise, print high score table and wait for START keypress.

```
600 IF CR>0 THEN BALL=CR+1:CR=0:POKE 6
56,2:POKE 657,11:? "0 ";:GOTO 285
610 SOUND 1,0,0,0:SOUND 2,0,0,0:FOR I=
1 TO 300:NEXT I
615 IF T>0 THEN T=0:A=USR(1536)
620 IF SC>H(10) THEN HS=SC:GOSUB 400:G
OSUB 675:GOTO 635
630 GRAPHICS 1:GOSUB 690:GOSUB 675
635 POKE 656,1:POKE 657,11:? "YOUR SCO
RE - ";SC
640 POSITION 4,19:? #6;" game  OVER ";
:FOR J=1 TO 30:GOSUB 660:NEXT J:POSITI
ON 4,19
650 ? #6;"PRESS  start";:FOR J=1 TO 30
:GOSUB 660:NEXT J:GOTO 640
660 IF PEEK(53279)<>6 THEN RETURN
670 GRAPHICS 5:GOSUB 690:GOTO 25
675 ? #6;"}":POSITION 5,2:? #6;"hi sco
res":FOR I=1 TO 10:POSITION 4,3+I:? #6
;H$(I*3-2,I*3);"  ";H(I):NEXT I
680 ? "}":RETURN
690 SETCOLOR 0,3,4:SETCOLOR 1,9,8:SETC
OLOR 2,12,4:POKE 752,1:RETURN
```

Title page graphics.

```
700 TRAP 750:X=0:Y=8:RESTORE 800:C=1
705 POKE 656,3:POKE 657,14:? "PRESS  S
TART";
710 READ J:IF J=0 THEN Y=Y+1:X=0:GOTO
710
720 C=1-C:IF C=0 THEN X=X+J:GOTO 710
725 IF PEEK(53279)=6 THEN RETURN
730 FOR I=0 TO J:PLOT X+I,Y:DRAWTO 39,
39:COLOR 0:PLOT X+I,Y:DRAWTO 39,39:COL
OR C:PLOT X+I,Y:POKE 53279,0:NEXT I
740 X=X+J:GOTO 710
750 POKE 656,0:POKE 657,18:? "By:":POK
E 656,1:POKE 657,13:? "Greg Schroeder"
760 FOR J=1 TO 3:FOR I=15 TO 0 STEP -0
.3:SOUND 0,150,8,I:SOUND 1,255,6,I:POK
E 708,INT(I*15)+4:NEXT I:NEXT J
775 SETCOLOR 0,3,4:D=PEEK(560)+256*PEE
K(561):X=PEEK(D+4):Y=PEEK(D+5):X1=X:Y1
=Y
780 FOR I=0 TO 3:X1=X+20*I:IF X1>255 T
HEN X1=(X+20*I)-256:Y1=Y1+1
782 POKE D+4,X1:POKE D+5,Y1:FOR J=1 TO
 7:GOSUB 798:NEXT J:NEXT I:FOR I=1 TO
2:X1=X+60-I*20
784 IF X1<0 THEN X1=256+(X+60-20*I):Y1
=Y1-1
786 POKE D+4,X1:POKE D+5,Y1:FOR J=1 TO
 7:GOSUB 798:NEXT J:NEXT I:GOTO 780
798 IF PEEK(53279)<>6 THEN RETURN
799 POP :POKE D+4,X:POKE D+5,Y:RETURN
```

Title page data.

```
800 DATA 12,5,2,5,2,5,2,5,2,5,2,1,5,1,
2,1,2,1,4,1,2,1,0,12,1,6,1,3,1,2,1,3,1
,2,1,6,1,3,1,2,2,3,2,2,1,2,2,3,1,2,1,0
```

SCREAMIN' DEMON

```
810 DATA 12,1,6,1,6,1,3,1,2,1,6,1,3,1,
2,1,1,1,1,1,1,1,2,1,2,1,1,1,2,1,0,12,5
,2,1,6,5,2,4,3,5,2,1,2,1,2,1,2,1,2,1
820 DATA 2,1,1,1,0,16,1,2,1,6,1,1,1,4,
1,6,1,3,1,2,1,5,1,2,1,2,1,3,2,0,16,1,2
,1,3,1,2,1,2,1,3,1,6,1,3,1,2,1,5,1,2,1
830 DATA 2,1,4,1,0,12,5,2,5,2,1,3,1,2,
5,2,1,3,1,2,1,5,1,2,1,2,1,4,1,0,0,0,0,
0,0,19,4,3,5,2,1,5,1,2,6,2,1,4,1,2,1,0
840 DATA 19,1,3,1,2,1,6,2,3,2,2,1,4,1,
2,2,3,1,2,1,0,19,1,3,1,2,1,6,1,1,1,1,1
,1,1,2,1,4,1,2,1,1,1,2,1,2,1,0
850 DATA 19,1,3,1,2,4,3,1,2,1,2,1,2,1,
4,1,2,1,2,1,1,1,2,1,0,19,1,3,1,2,1,6,1
,5,1,2,1,4,1,2,1,3,2,2,1,0
860 DATA 19,1,3,1,2,1,6,1,5,1,2,1,4,1,
2,1,4,1,0,19,4,3,5,2,1,5,1,2,6,2,1,4,1
,2,1,0,X
870 DIM X$(78):FOR I=1 TO 78:READ S:PO
KE 1535+I,S:X$(I,I)=CHR$(S):NEXT I:S=1
:RETURN
```

SWAT TABLE

For ATARI® SCREAMIN' DEMON

LINES	SWAT CODE	LENGTH
1 - 6	JJ	569
8 - 60	YY	514
65 - 90	MF	565
92 - 102	YJ	555
103 - 115	WI	539
116 - 160	UP	475
180 - 217	VB	440
220 - 285	XO	502
290 - 305	AG	551
307 - 342	ZE	546
345 - 355	RA	637
357 - 420	RX	548
425 - 460	QT	501
465 - 485	BL	643
490 - 505	MB	566
510 - 530	QH	531
535 - 560	TO	558
564 - 575	TQ	663
576 - 595	DQ	608
600 - 650	GM	552
660 - 720	UL	547
725 - 775	PH	519
780 - 800	AX	521
810 - 850	GZ	533
860 - 870	UI	168

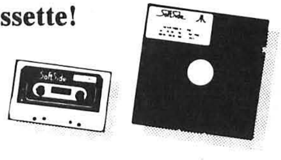
Tired of Typing?

- Use And Enjoy These Programs Right Away!
- Avoid Worry About Typos!
- Get Additional Programs to Enjoy Without Typing.

Order this Issue's Programs on Disk or Cassette!

Send the coupon below to:

SoftSide Publications, Inc.
10 Northern Blvd.
Northwood Executive Park
Amherst, NH 03031



SoftSide CV SoftSide DV

SoftSide™ Selections

ORDER THIS ISSUE'S MEDIA

Having difficulty finding the time to input the programs from this issue of ***SoftSide* Selections**? Don't despair — the cassette and disk versions are still available! **And**, each DV (disk version) contains an additional program.

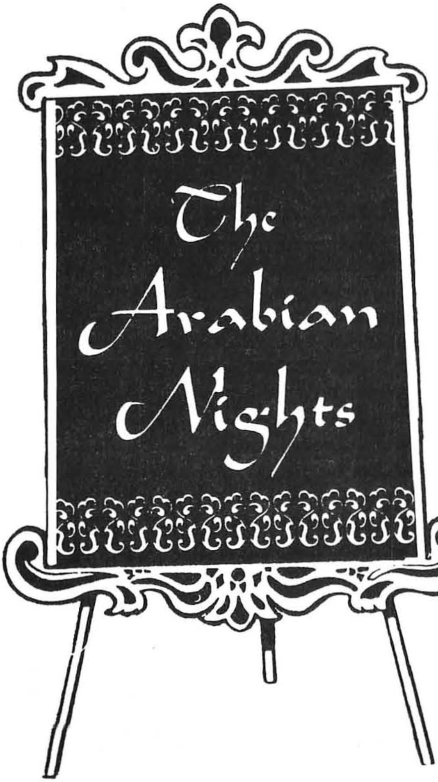
□ **#42 DV $16.00** □ **#42 CV $8.00**

Disk Only: □ Apple® □ IBM® PC
Disk or Cassette: □ Atari® □ TRS-80® Mod. I □ TRS-80® Mod. III

Name ____________________
Address ____________________
City/State ______________ Zip ________
□ **Check or Money Order**
□ MasterCard □ VISA
Name of Cardholder ____________________
MC# or Interbank#/VISA# ____________________
Exp. Date ________ Signature ____________________

Prices subject to change without notice. Apple®, IBM®, Atari® and TRS-80® are registered trademarks of The Apple Computer Company, International Business Machines, Inc., Warner Communications, and The Tandy Corporation respectively.

SoftSide™ ADVENTURE SERIES



Issue 42 Adventure: Arabian Nights

The daughter of the Caliph of Baghdad lies in a death-like trance, the victim of the evil magician Roxor's spell. Scattered throughout the desert are the tools and magical items you will need to acquire the eggshell of the great Roc, which can restore health to the Princess. Beware of monsters and poisoned water, and heed well the advice of the seer! Good luck, sahib!

SoftSide Adventure Series CV DV

What would you say to a program that asks, "What do you want to do?" Well, you might say, "GET RUBY" or "KILL GIANT", because that's how the *SoftSide Adventure Series* works.

Each issue, the latest Adventure takes you to another world of fantasy, puzzles, and thrills. Your first task is often simple survival — and even that basic feat can be daunting until you figure out the *right* way to do it. You'll have to be ingenious and persevering, and your rewards will be great.

To "win" a fantasy/adventure game, you must solve the author's devious puzzles, and overcome the obstacles that confront you — whether they be dragons or desperadoes. Death, should it come, is transitory — just re-run the program to live again!

Experienced adventurers create detailed maps of each world as they search for solutions, but you can omit this exercise if your memory is exceptional. Express your wishes with one- or two-word commands, like "LOOK", "DROP KNIFE", or "GET RUBY". Use "I" to get an inventory of your possessions. This issue's Adventure, *Arabian Nights*, features the commands "SAVE GAME" and "LOAD GAME". These permit you to try to solve the Adventure in more than one sitting. You'll also want to save the game before trying something hazardous, so that you may resume the game, should the results be adverse, without having to repeat a lot of work. As always, use the command "HINT" to decode the hints we publish one issue after each Adventure's appearance. The introduction to each Adventure explains this more fully.

To start up the Adventure, just run the program called "INTRO" or "INTRO.BAS" on your disk, or select the Adventure from the DV menu.

On cassette, the INTRO program is the one just before the Adventure, which is the last program on the tape.

The Adventure runs on any Atari with at least 32K RAM (40K disk).

Here are the encrypted hints for *Atlantis,* the Adventure in issue 41.

To find Atlantis: ULOOLD GSV NVINZRW.

If something you need seems to be missing: OLLP RM GSV QFMP LI ZG IVGVK DSVM SV'H WVZW.

What to do with the coins: GSILD GSVN RMGL GSV ULFMGZRM.

What to do with the rods: RMHVIG GSVN IZGSVI GSZM WILK GSVN.

To open the foot locker: WVHGILB RG DRGS GSV YOZHGVI.

The injured Atlantean: TL YZXP GL DSVIV SV DZH SFIG ZUGVI BLF'EV SVOKVW SRN.

If you get thrown into jail a second time: TREV FK --BLF XZM'G VHXZKV UILN QZRO GDRXV!

General Information About Listings, SWAT, and Magnetic Media

These are the standard procedures for the programs published in **SoftSide Selections.** Sometimes, a particular program does not lend itself to these procedures. Always read the specific instructions accompanying a program. They will instruct you if there are any variances from the following procedures. Also, back issues of **SoftSide** Magazine may differ in some details.

SWAT Tables

At the conclusion of each program listing in **SoftSide Selections**, we include a **SWAT (Strategic Weapon Against Typos)** Table. **SWAT** for the Atari appeared in **SoftSide** Issue #30. If you missed Issue #30, we'll send you a free reprint of **SWAT.** Send a self-addressed, stamped envelope to: **SoftSide** Publications, Inc.
Department **SWAT**
10 Northern Blvd.
Northwood Executive Park
Amherst, NH 03031

Be sure to tell us that you have an Atari computer.

Magnetic Media

Disks do not carry the DOS.SYS and DUP.SYS files, and are not "bootable." First, boot a disk with DOS on it, then insert the **SoftSide Selections** disk, and run "D:COVER". Our disks are in DOS 2 format.

Tapes CLOAD in the normal manner. If you encounter difficulty, try this procedure:

1. POKE 54018,54
2. Turn up the volume on your TV.
3. Type CLOAD, and press RETURN once.
4. Press the play button, and listen.
5. When you hear the steady leader tone, press RETURN again.

Side two of the tape is a duplicate of side one.

SoftSide Selections disks and tapes are duplicated on reliable, professional equipment. Bad copies are exceedingly rare. Nevertheless, the trip through the mail occasionally results in damage to the sensitive magnetic media. If, after a reasonable number of attempts on well-adjusted, clean equipment, you are unable to load a program, return it to us along with an exact explanation of your problem. We will send you a replacement.

SoftSide Selections media are not copy protected. We urge you to make an archival backup copy of your disk or tape as soon as you receive it, as our replacement policy is valid only for 30 days. Please resist the urge to give away copies of copyrighted material.

Line Listings

Line listings are in standard 38-column format, with special conventions for representing unprintable characters:

You must type underlined characters, including blank spaces, in inverse video.

When graphics or control (CTRL) characters are included in a string (between quotation marks), a nearby REM statement will make note of it; in such cases, graphics characters appear as the corresponding lower-case letters, and control characters appear as the corresponding unshifted key symbols. For example: the lower-case letter **s** represents a graphic cross, which you type by pressing the S key while holding down the CTRL key; the = sign represents CTRL-down-arrow, which you type by pressing and releasing the ESC key, then pressing the = key while holding down CTRL. For more information about entering control characters, refer to Appendix F and the back cover of your **Atari BASIC Reference Manual.**

There are two exceptions to our above convention: A clear-screen character (ESC SHIFT-CLEAR) appears in our listings as a right-hand brace, which looks like this: }. The other exception is that a shifted = sign appears as a broken vertical line: ¦ .

Occasionally, a program will demand that we vary from these conventions. In such a case, a nearby REM statement or the program's introductory article will clearly note the special instructions.

Be sure to read each program's explanatory article — it may contain special, important information about the program. Also, use **SWAT** on your program, and get the free reprint if you don't have **SWAT.**

System Requirements

The necessary memory and other equipment you need to run a program are listed in the introductory paragraph of the article for each program. (Also see the **SoftSide Adventure Series** elsewhere in this booklet.)

SoftSide™ Selections

Here's **SoftSide Selections**, the handy, pull-out booklet with program listings for your Atari® 400/800/1200 computer. This issue, **SoftSide Selections** for the Atari features:

- **Microtext 2.0** — This issue's Front Runner is a BASIC word processor with editing, printout, and even search commands, plus disk storage of your text.

- **Screamin' Demon** — Enjoy pinball action with this simulation of gleaming steel balls, buzzers, bells, and digit counters.

- **Atari DV Bonus Program: Family Tree Organizer** — a sophisticated, multi-program system that can keep detailed records of your family's genealogy and history.

- **The SoftSide Adventure Series — Arabian Nights.** The daughter of the Caliph of Baghdad lies in a death-like trance, the victim of the evil magician Roxor's spell.

SOFTSIDE PUBLICATIONS, INC.
Amherst, New Hampshire