SoftSide®

FRONT RUNNER

Printed Game Software for Atari®, Apple®, IBM® PC.

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

SoftSide®

FRONT RUNNER

Index

CRISIS by Lyle Grant........................ **2**
A ruthless race of aliens has taken possession of the Empire State Building. Your goal is to rescue the hostages.

KANGARILLA by Oscar Bascara............... **7**
IBM® PC version by Kerry Shetline
Your baby kangaroo has wandered to the top of a four-level maze, and you must rescue him.

REBOUND by Steven Kuehne............... **11**
Fire the moving ball into the colored goal in the center of your screen, but count the barriers you erect with each press of the joystick. They block or bounce your ball in the right direction — and lower your score as they accumulate.

GENERAL INFORMATION
and SYSTEM REQUIREMENTS.............. **18**

This booklet prepared by the staff of SoftSide Publications, Inc.,

CRISIS

by Lyle Grant

Crisis is an arcade-style color graphics game for any Apple® II computer with one disk drive and 48K RAM.

A ruthless race of aliens has captured the Empire State Building, and holds the people inside as hostages. Your task is to rescue as many of the hostages as you can. You have three "men" to perform this feat of daring and bravery. The aliens have taken measures to prevent such a rescue. On each floor of the building, you encounter a poison cloud, an electric beam, and walls that suddenly appear in your path.

To free a hostage, you must touch him as you move about the floor. Use the I, J, K and M keys to move up, left, right and down. If you take too long to free a prisoner, he moves to another spot on the floor. Each time you free a hostage, you earn 100 points. If you save eight hostages without being killed, a door opens up on the left side of the screen. Going through the door gets you bonus points and a ride to the next floor. There you must rescue another eight, but the game gets progressively faster and harder.

Variables

A, B: Electric beam's position.

A(*): Stored machine language data.

A1, A2, X, X$: Miscellaneous variables.

B: Bonus points for finishing a floor.

B(*): Location of the cloud's and beam's machine language movement data.

C(*): Positions of new walls.

DR: Tells when the door has opened.

EE, MM, NN: Position and shape of the hostage.

HS: The number of hostages left on the floor.

J, K: The cloud's position.

KA, KB, KC, KD, MT, VK, VL, VM, VN, VP: Constants.

KY: Keyboard input.

LV: Floor number.

MA: Time since the hostage last moved.

MB: Time between relocations of the hostage.

MN: Lives left.

SC: Score.

VJ: Width of door.

WA, WL: Length of new walls.

WG: Time since the walls last appeared.

WH: Time between appearances of walls.

XX, YY: Player's position.

Z: Player's direction.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                            SS
SS       Applesoft BASIC      SS
SS           'Crisis'         SS
SS      Author: Lyle Grant    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 #46 SoftSide DV.

Main program.

```
100  GOTO 680
110 KY =  PEEK (KD):KY = KY - 32 *
     (KY > 95): IF KY > KA AND KY
      < KC AND KY <  > KB THEN Z =
     KY - KA
120  COLOR= 4: PLOT XX,YY
130  ON Z GOSUB 330,340,350,360,3
     70
140  IF  SCRN( XX,YY) = 15 THEN  GOSUB
     660: GOSUB 460: GOSUB 600
150  IF  SCRN( XX,YY) <  > 4 THEN
     940
160  COLOR= 7: PLOT XX,YY
170  FOR X = 1 TO 4
180 A(X) =  PEEK (B(X)):B(X) = B(
     X) + 1: NEXT
190 A = A(1):J = A(2):B = A(3):K =
     A(4)
200  IF B(1) = 31821 THEN B(1) =
     31750:B(3) = 31825
210  IF B(2) = 31979 THEN B(2) =
     31900:B(4) = 31980
220  IF XX >  = J - 1 AND XX <  =
     J + 3 AND YY >  = K - 1 AND
     YY <  = K + 3 THEN  GOSUB 94
     0
230  COLOR= 13: VLIN A,A + .5 AT
     B
240  COLOR= 11: FOR X = 1 TO 2
250  VLIN K,K + 3 AT J: VLIN K,K +
     3 AT J + 1: VLIN K,K + 3 AT
     J + 2
260  COLOR= 4: NEXT
270  PLOT A,B
280 WG = WG + 1: IF WG = WH THEN
      GOSUB 490
290 MA = MA + 1: IF MA = MB THEN
      COLOR= 4: GOSUB 660: GOSUB
     620
300  GOSUB 650
310  IF HS <  = 0 THEN  GOSUB 380
320  GOTO 110
```

Player's movement.

```
330 YY = YY - MT: RETURN
340 XX = XX - MT: RETURN
350 XX = XX + MT: RETURN
360  POP : GOTO 110
370 YY = YY + MT: RETURN
```

Routine to handle open door.

```
380 DR = DR + 1: IF DR = 1 THEN  GOSUB
     420
390  COLOR= 4: VLIN VJ,VK AT VL
400  IF YY <  = VK + 1 AND YY >  =
     VJ AND XX < VM THEN 890
410  RETURN
```

Sound for open door.

```
420  FOR X = 1 TO 700: NEXT : FOR
     X = 1 TO 5
430  & 28,20: & 30,20: & 38,15
440  NEXT
450  & 28,50: RETURN
```

Draw border walls.

```
460  COLOR= 13
470  HLIN VL,VN AT VL: HLIN VL,VN
      AT VN: VLIN VL,VN AT VL: VLIN
     VL,VN AT VN: HLIN VL,VN AT V
     P
480  RETURN
```

Draw new walls.

```
490  COLOR= 0
500  FOR X = 1 TO 4
510 RN =  RND (1) * 3): IF (X = 2
      AND  ABS (RN - YY) < 4) OR
     (X = 4 AND  ABS (RN - XX) <
     4) THEN 510
520 C(X) = RN: NEXT
530 WL = C(1) + 7:WA = C(3) + 7
540  FOR X = 1 TO 2
550  HLIN C(1),WL AT C(2)
560 C(2) = C(2) + 1: NEXT
570  VLIN C(3),WA AT C(4)
580  GOSUB 460
590 WG = 0: RETURN
```

Hostage saved.

```
600  & 30,15: & 28,8: & 15,10
610 SC = SC + 100: IF HS >  = 1 THEN
     HS = HS - 1
620  VTAB 21: HTAB 14: PRINT SC: HTAB
     39: VTAB 22: PRINT HS
```

Draw new hostage.

```
630 MM =  RND (1) * 37:EE = MM +
     2:NN =  RND (1) * 36
640 MA = 0: GOSUB 460
650  COLOR= 15
660  HLIN MM,EE AT(NN + 1): VLIN
     NN,NN + 2 AT MM + 1: PLOT MM
     ,NN + 3: PLOT EE,NN + 3
670  RETURN
```

Initialization.

```
680  DIM A(4): DIM B(4): DIM C(4)
690  GOSUB 1070
700  READ MT,KA,KB,KC,KD,VJ,VK,VL
     ,VM,VN,VP
710  GOSUB 840
720 LV = 1:MN = 3:WH = 33:MB = 60
     :SC = 0
730 B(1) = 31750:B(2) = 31900:B(3
     ) = 31825:B(4) = 31980
740  GR : GOSUB 1050
750 XX = 18:YY = 19: COLOR= 6: PLOT
     XX,YY
760 HS = 8
770  GOSUB 490: GOSUB 490: GOSUB
     460: GOSUB 630
780  POKE 49168,0:Z = 0: HOME
790  VTAB 21: HTAB 8: PRINT "Scor
     e:"SC: HTAB 21: VTAB 22: PRINT
     "Hostages on floor:"HS
800  VTAB 21: HTAB 24: PRINT "Flo
     or:"LV
810  VTAB 22: HTAB 3: PRINT "Live
     s:"MN
820  FOR X = 1 TO 800: NEXT X
830  GOTO 110
```

Instructions.

```
840  HOME : HTAB 14: PRINT "--INS
     TRUCTIONS--"
850  VTAB 5: PRINT "Movement keys
     :  'I' = up": PRINT : HTAB 1
     7: PRINT "'M' = down"
860  PRINT : HTAB 17: PRINT "'J'
     =left": PRINT : HTAB 17: PRINT
     "'K' =right"
```

```
870  VTAB 14: PRINT "Scoring: hos
     tages = 100 pts. each.": PRINT
     : PRINT "Bonus pts. given af
     ter finishing a floor"
880  VTAB 23: HTAB 9: PRINT "-Pre
     ss SPACE to begin-";: GET X$
     : RETURN
```

Set up new floor.

```
890  & 50,25: & 48,15: & 40,25: &
     38,15: & 30,25: & 28,15: & 2
     0,25: & 15,25
900  TEXT : HOME : VTAB 11: HTAB
     10:B = 1000 * LV: PRINT "Bon
     us="B;: PRINT " pts."
910 LV = LV + 1:WH = WH - 1:MB =
     MB - 3:DR = 0:VJ = VJ + 1
920 SC = SC + B: VTAB 22: HTAB 4:
      PRINT "Score:"SC
930  FOR X = 1 TO 1000: NEXT X: GOTO
     730
```

Player loses one life.

```
940  & 255,100: & 250,100: & 220,
     40: & 200,150: & 190,255: &
     240,255: & 254,255
950 MN = MN - 1
960  IF MN <  > 0 THEN 730
```

Game over. Ask player for another game.

```
970  INVERSE : PRINT ">>>>>>>>>>>>
     >>>>GAME OVER<<<<<<<<<<<<<<
     <";: NORMAL
980  POKE 49168,0
990  FOR X = 1 TO 1000: NEXT
1000  PRINT "Do you want to play
     again? ";
1010  GET X$:AX =  ASC (X$): IF A
     X > 95 THEN AX = AX - 32
1020  IF AX <  > 78 AND AX <  > 8
     9 THEN 1010
1030  IF AX = 78 THEN  TEXT : HOME
     : END
1040  GOTO 720
```

Draw background color.

```
1050  FOR X = 1 TO 38
1060  COLOR= 4: HLIN VL,VN AT X: NEXT
     : RETURN
```

Display title page.

```
1070  PRINT  CHR$ (21);: TEXT : HOME
     : NORMAL : VTAB 8: HTAB 15: PRINT
     "++CRISIS++": HTAB 19: VTAB
     17: PRINT "by": VTAB 19: HTAB
     15: PRINT "Lyle Grant"
```

Poke in sound routine and machine language data.

```
1080  FOR X = 31750 TO 31821: READ
     A1,A2: POKE X + 75,A2: POKE
     X,A1: NEXT
1090  FOR X = 31900 TO 31978: READ
     A1,A2: POKE X + 80,A2: POKE
     X,A1: NEXT
1100  FOR X = 768 TO 833: READ A1
     : POKE X,A1: NEXT
1110  POKE 1013,76: POKE 1014,0: POKE
     1015,3
1120  RETURN
```

Sound routine and machine language data.

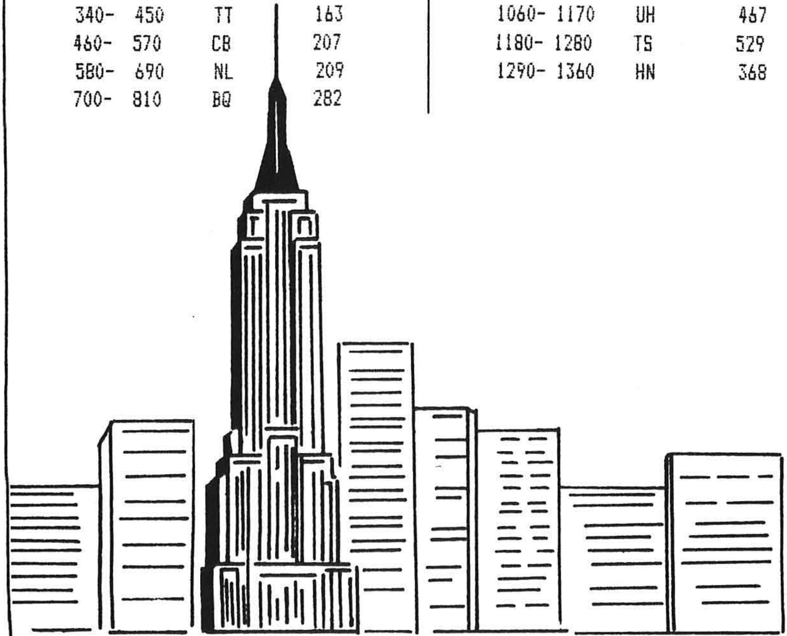
```
1130  DATA 20,2,19,3,18,4,17,5,16
     ,6,15,7,14,8,13,9,12,10,11,1
     1
1140  DATA 10,12,9,13,8,14,7,15,6
     ,16,5,17,4,18,3,19,2,20,3,21

1150  DATA 4,22,5,23,6,24,7,25,8,
     26,9,27,10,28,11,29
1160  DATA 12,30,13,31,14,32,15,3
     3,16,34,17,35,18,36
1170  DATA 19,37,20,38,21,37,22,3
     6,23,35,24,34,25,33,26,32
1180  DATA 27,31,28,30,29,29,30,2
     8,31,27,32,26
1190  DATA 33,25,34,24,35,23,36,2
     2,37,21,38,20,37,19,36,18
1200  DATA 35,17,34,16,33,15,32,1
     4,31,13,30,12,29,11,28,10
1210  DATA 27,9,26,8,25,7,24,6,23
     ,5,22,4,21,3
1220  DATA 2,29.3,26,3,24,4,23,4,
     21,4,19,5,16,5,14,6,11
```

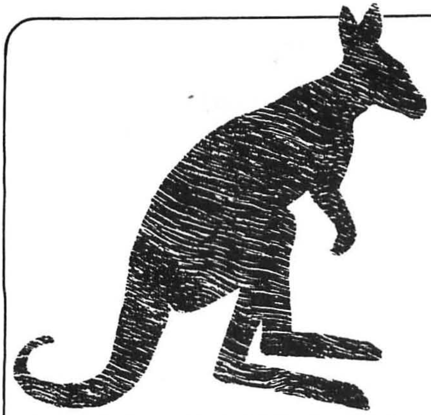
```
1230  DATA 6,9,7,6,7,4,8,2,9,5,10
     ,8,10,10,11,13,11,15,12,18
1240  DATA 13,21,13,23,14,26,14,2
     8,15,30,16,33,17,35,18,32,19
     ,29
1250  DATA 20,26,21,23,22,20,23,1
     7,24,14,25,11,26,8,27,6
1260  DATA 27,5,28,2,29,4,30,7.31
     ,10,32,13,33,16,34,19,35,22
1270  DATA 36,25,35,26,34,27,33,2
     8,32,29,31,30,30,31,29,32
1280  DATA 28,33,27,34,26,35,25.3
     2,24,29,23,26,22,23,21,20
1290  DATA 20,17,19,14,18,11,17,8
     ,16,5,15,2,14,5,13,8,12,11
1300  DATA 11,14,10,17,9,20,8,23,
     7,26,6,29,5,32,4,35,3,32
1310  DATA 201,84,206,15,32,177,0
     ,32,248,230,138,72,32,183
1320  DATA 0,201,44,240,3,76,201,
     222,32,177,0,32,248,230
1330  DATA 104,134,3,134,1,133,0,
     170,160,1,132,2,173,48
1340  DATA 192,136,208,4,198,1,24
     0,7,202,208,246,166,0,208
1350  DATA 239,165,3,133,1,198,2,
     208,241,96
1360  DATA 1.2,200,204,206,49152,
     7,22,0,2,39,1
```

STOMP TABLE

For **Apple® CRISIS**

LINES	STOMP CODE	LENGTH
100- 210	ZH	287
220- 330	XR	197
340- 450	TT	163
460- 570	CB	207
580- 690	NL	209
700- 810	BQ	282
820- 930	UM	454
940- 1050	LF	270
1060- 1170	UH	467
1180- 1280	TS	529
1290- 1360	HN	368





KANGARILLA

by Oscar Bascara
Translation by Kerry Shetline

Kangarilla is an arcade-style game for an IBM® PC with disk drive, 64K RAM and color graphics adapter. A joystick controller is optional.

In *Kangarilla,* you control a distraught mother kangaroo whose baby is stranded at the top of the four-level maze. To reach the upper levels, you must make the kangaroo leap through holes in the ceiling.

The program asks whether you wish to use the keyboard or a joystick. If you choose joystick control, move the joystick to control her direction, and use either button to make her jump. Be sure to press the button when the kangaroo's foot is on the ground. On the keyboard, use the cursor keypad: left-arrow for left, down-arrow to stop, right-arrow for right, and up-arrow to jump.

When you make the kangaroo leap, be careful not to make her bump into the ceiling, and watch out for the rolling ball that tries to trip her up. If she hits the ceiling or ball, the round is over, and you move to your next kangaroo.

If the kangaroo reaches her baby, the baby is suddenly transported away, but if you succeed six times, mother and baby live happily ever after. You will need good timing and a bit of strategy to accomplish this. Are you up to the *Kangarilla* challenge?

Variables

A$: General input string.
BALL%: Graphics array for ball.
D(*): The kangaroo's change in position, according to the direction of movement.
FL: A flag that determines whether to make sound.
G: The number to add to the kangaroo's y-coordinate.
IT: The shape number of the kangaroo.
KANG1% — KANG4%: Graphics arrays for kangaroo.
K,K2: ASCII values of input.
L: The y-coordinate of the level.
M: The ball's horizontal coordinate.
MO: Mode of operation — 1 for keyboard, 2 for joystick.
P: Pause loop variable.
Q: Level number.
SHAPE$(*): Kangaroo shapes.
SP: Board number — controls speed of ball.
T: Contains 0, 1, or 2, depending on the direction of movement.
U: Number of kangaroos used.
X, Y: Coordinates of the kangaroo.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                            SS
SS     PC Advanced BASIC      SS
SS         'Kangarilla'       SS
SS  Author: Oscar C. Bascara  SS
SS Translator: Kerry Shetline 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 #46 SoftSide DV.

Initialization.

```
100 KEY OFF: SCREEN 1,0: CLS
110 DIM BALL%(5),KANG1%(11),KANG2%(12),K
ANG3%(11),KANG4%(12),SHAPE$(4): DEF FNR(
X)=INT(RND(1)*X): DEF FNUP$(X$)=CHR$(ASC
(X$)+32*(ASC(X$)>96))
120 FOR X=1 TO 4: READ SHAPE$(X): PRESET
 (X*20,0): DRAW "C3"+SHAPE$(X): NEXT
130 DATA "G1 L1 G1 R2 D1 L1 G1 R2 BR3 BD
1 L1 BL2 L1 G1 R4 G1 L3 D1 R2 BD1 BL2 L1
"
140 DATA "G1 L1 G1 R2 D1 L1 G1 R2 BR3 BD
1 L1 BL2 L1 G1 R4 G1 L3 D1 R2 F1 G1"
150 DATA "BL2 F1 R1 F1 L2 D1 R1 F1 L2 BL
3 BD1 R1 BR2 R1 F1 L4 F1 R3 D1 L2 BD1 BR
2 R1"
160 DATA "BL2 F1 R1 F1 L2 D1 R1 F1 L2 BL
3 BD1 R1 BR2 R1 F1 L4 F1 R3 D1 L2 G1 F1"
170 PSET (10,0): DRAW "R1 F1 L3 D1 R3 G1
 L1"
180 BABY$="F1 R1 F1 L2 D1 R1 D1 L1 D1 R2
"
190 GET (9,0)-(12,3),BALL%: GET (16,0)-(
23,9),KANG1%: GET (36,0)-(43,10),KANG2%:
 GET (56,0)-(63,9),KANG3%: GET (76,0)-(8
3,10),KANG4%: CLS
200 RANDOMIZE VAL(MID$(TIME$,3,2))*100+V
AL(RIGHT$(TIME$,2))
```

Prompt user for control mode.

```
210 LOCATE 7: PRINT "Joystick or Keyboar
d? (J/K)": LINE (224,48)-(231,55),,BF
220 A$=INKEY$: IF A$="" THEN 220 ELSE A$
=FNUP$(A$): IF A$="J" THEN MO=2: STRIG O
N ELSE IF A$="K" THEN MO=1 ELSE 220
230 CLS
240 SP=1: D(0)=-2: D(1)=0: D(2)=2
```

Plot levels on the screen and place a randomly located hole in each level.

```
250 FOR I=230 TO 240 STEP 10: PSET (I,14
0): DRAW SHAPE$(1): NEXT
260 FOR I=135 TO 45 STEP -30: LINE (0,I)
-(279,I)
270 IF I=135 THEN 300
280 H=FNR(200)+4: IF POINT(H,I+30)=0 OR
POINT(H+15,I+30)=0 OR (I=45 AND H<30) TH
EN 280
290 LINE (H,I)-(H+15,I),0
300 NEXT
```

Draw the baby kangaroo. Draw a ball at the bottom of the screen to indicate board number.

```
310 PSET (1,39): DRAW BABY$: PUT (SP*5+5
,145),BALL%
```

Clear the keyboard. Initialize the kangaroo's coordinates, level, shape and "bounce formula."

```
320 DEF SEG=0: POKE 1050,PEEK(1052): X=2
60: Y=120: L=125: IT=1: G=0
```

Initialize ball coordinates.

```
330 M=9: PUT (M,L+6),BALL%
```

Beginning of main program loop. Read kangaroo's movement.

```
340 GOSUB 550
```

Bounce formula.

```
350 G=G+.5: X=X+D(T): Y=Y+G
```

Set shape number according to the direction and height of the kangaroo.

```
360 IF T<>1 THEN IT=T+1 ELSE IT=IT-((IT+
1) MOD 2)
370 IF ABS(G)<>G THEN IT=IT+1
380 GOSUB 1040
```

Make sure the kangaroo doesn't leave the side of the screen.

```
390 IF X>272 OR X<12 THEN GOSUB 1040: X=
X-D(T): GOSUB 1040
```

Check if the kangaroo's feet are on the ground. If so, set up a new bounce.

```
400 IF Y=L THEN G=-3: GOTO 450
```

Move the ball.

```
410 GOSUB 830
```

Check if the ball is hitting the kangaroo. If so, the kangaroo trips.

```
420 IF L+5<Y+9 AND L+5>Y AND M-(D(INT(IT
/3))*2)-7<X AND M-(D(INT(IT/3))*2)+2>X T
HEN GOSUB 740: GOTO 780
```

Check if the kangaroo's head hit the ceiling. If so, the kangaroo falls.

```
430 Q=L+10: IF INT(Y)=L+11 AND POINT(X,Q
)=3 THEN 620
```

Erase the kangaroo and go back to the beginning of the main loop.

```
440 GOSUB 1040: GOTO 340
```

Check if the kangaroo has rescued the baby.

```
450 IF L<50 AND X<40 THEN 970
```

Check if the kangaroo will fall through a hole.

```
460 IF T=1 THEN T=0
470 Z=X+D(T)-1: IF Q<135 AND T=2 AND POI
NT(X+5,Q)=0 THEN Z=X+5
480 IF Q<135 AND T=2 AND POINT(X+3,Q)=0
THEN Z=X+3
490 IF Q<135 AND POINT(Z,Q)=0 AND FL=0 T
HEN G=1.5: E=-30: GOSUB 540: GOTO 410
```

Produce bouncing sound unless the kangaroo is falling through a hole.

```
500 IF FL=0 THEN SOUND 700,.1
510 FL=0
```

Check for a jump.

```
520 GOSUB 590
```

Go back to the main loop.

```
530 GOTO 410
```

Place the ball on the same level as the kangaroo.

```
540 PUT (M,L+6),BALL%: L=L-E: PUT (M,L+6
),BALL%: RETURN
```

Read direction from keyboard or joystick.

```
550 IF MO=2 THEN T=INT(STICK(0)/90): RET
URN ELSE A$=INKEY$: IF A$="" THEN T=HT:
RETURN
560 IF LEN(A$)=1 THEN A$=FNUP$(A$): K=AS
C(A$): K2=0 ELSE K=0: K2=ASC(MID$(A$,2))
570 IF K=52 OR K2=75 THEN T=0 ELSE IF K=
54 OR K2=77 THEN T=2 ELSE T=1
580 HT=T: RETURN
```

Check for jump, keyboard or joystick control.

```
590 IF L<51 THEN RETURN ELSE IF MO=1 THE
N IF K=32 OR K2=72 THEN 610 ELSE RETURN
600 IF NOT STRIG(1) AND NOT STRIG(5) THE
N RETURN
610 G=-6: E=30: GOSUB 540: FL=1: K=0: K2
=0: RETURN
```

Kangaroo falls after hitting her head on the ceiling.

```
620 GOSUB 740
630 IF IT=4 THEN DRAW "A1": FD=36 ELSE D
RAW "A3": FD=38
640 FOR I=Y+5 TO L+FD
650 PSET (X,I): DRAW SHAPE$(IT)
660 FOR P=1 TO 20: NEXT
670 PRESET (X,I): DRAW SHAPE$(IT)
680 NEXT
690 FL=0
```

Check if the kangaroo was hit three times. If so the game ends.

```
700 U=U+1: IF U=3 THEN LOCATE 21,1: PRIN
T "Game Over.": GOTO 1020
```

Subtract one kangaroo from the indicator that shows how many kangaroos remain.

```
710 DRAW "A0"
720 PRESET (220+10*U,140): DRAW SHAPE$(1
)
730 GOTO 320
```

Make a sound, then clear the kangaroo and the ball from the screen.

```
740 SOUND 300,.1
750 GOSUB 1040
760 PUT (M,L+6),BALL%
770 RETURN
```

The kangaroo falls over.

```
780 IF D(T)=2 THEN IT=3: A=6 ELSE IT=1:
A=4
790 LINE (X-4,Y)-(X+3,Y+9),0,BF: DRAW "A
1"
800 PRESET (X-12,L+A): DRAW "C3"+SHAPE$(
IT)
810 FOR P=1 TO 750: NEXT: PRESET (X-12,L
+A): DRAW SHAPE$(IT)+"A0"
820 GOTO 700
```

Subroutine for moving the ball.

```
830 PUT (M,L+6),BALL%
840 M=M+SP: IF M>275 THEN M=5
850 PUT (M,L+6),BALL%
860 RETURN
```

The kangaroo reaches her baby.

```
870 GOSUB 750
880 LOCATE 7,1: PRINT "Mom!"
890 I=X: FOR X=I TO 11 STEP -1
900 GOSUB 1040
910 FOR P=1 TO 30: NEXT
920 GOSUB 1040: NEXT
930 GOSUB 1040
940 LOCATE 4,2: PRINT CHR$(3)
```

Set higher ball speed. If six boards are completed, the game is won.

```
950 SP=SP+1: IF SP=7 THEN LOCATE 21,1: P
RINT "You won!": GOTO 1020
960 FOR P=1 TO 1000: NEXT
970 PRESET (1,39): DRAW BABY$
980 LOCATE 7,4: PRINT "?"
990 LOCATE 4,2: PRINT "?"
```

Baby kangaroo floats away.

```
1000 FOR I=38 TO 0 STEP -1: PSET (1,I):
DRAW BABY$: FOR P=1 TO 30: NEXT: PRESET
(1,I): DRAW BABY$: NEXT I
```

Start the next board.

```
1010 FOR P=1 TO 1200: NEXT: LINE (0,0)-(
319,135),0,BF: GOTO 260
```

Check if the player wants another game.

```
1020 PRINT: PRINT "Would you like to pla
y again? (Y/N)": LINE (298,176)-(295,183
),,BF
1030 A$=INKEY$: IF A$="" THEN 1030 ELSE
A$=FNUP$(A$): IF A$="Y" THEN RUN ELSE IF
 A$="N" THEN CLS: END ELSE 1030
```

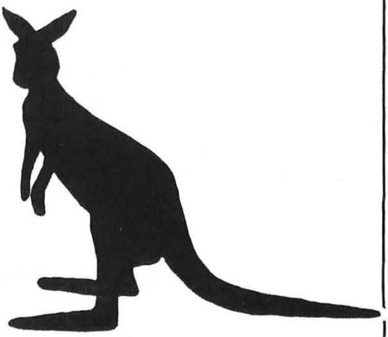
Kangaroo plotting routine.

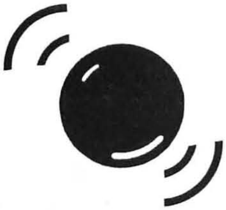
```
1040 XX=X-4: ON IT GOTO 1050,1060,1070,1
080
1050 PUT (XX,Y),KANG1%: RETURN
1060 PUT (XX,Y),KANG2%: RETURN
1070 PUT (XX,Y),KANG3%: RETURN
1080 PUT (XX,Y),KANG4%: RETURN
```

STOMP TABLE

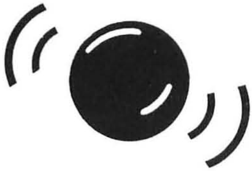
For IBM® PC KANGARILLA

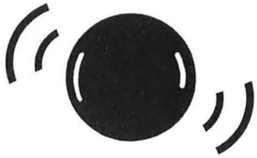
LINES	STOMP CODE	LENGTH
100 - 170	VN	511
180 - 290	WL	499
300 - 410	QI	251
420 - 530	GF	294
540 - 650	VL	357
660 - 770	IM	157
780 - 890	DF	232
900 - 1010	DC	227
1020 - 1080	LI	227





rebound





by Stephen Kuehne

Rebound is a two-player, abstract, arcade-style game for an Atari® with 16K RAM (24K with disk) and one or two joysticks.

Your joystick controls everything in *Rebound* — no need for the keyboard. If you have one joystick, plug it into port one; plug a second stick into port two. Use the joystick button to move past the instructions after you've read them. Then, move the stick up and down to select your responses to the set-up questions (one or two joysticks, skill levels, number of rounds to play). The button tells the Atari that you've made your choice.

The object of *Rebound* is to direct the moving ball into the colored goal at the center of the screen by placing diagonal barriers in the ball's path. You start play in each round by pressing the joystick button after the word "Serve" appears at the top of the screen. When you move your joystick left or right, a barrier appears. The type of barrier depends on the direction you move the stick. If you construct more than fifteen barriers, you can't receive any points, and if you use all your time, your opponent gets 25 bonus points. The amount of time you have depends on your skill level. A beeper warns when your time is almost up. If the ball goes into the goal and you've placed no barriers, it's a lucky serve, and you have to go again.

The game gets more complex as play progresses, because the barriers stay on the screen. Don't get too frustrated when you can't get the ball to go where you want — it's only a game.

Variables

A$: Used to hold graphics.

AT: Allowed time in which to finish a round. TT counts up to AT by one for each movement of the ball.

CHBAS: This times 256 is the low end of the transferred character set.

CHROM: This times 256 is the low end of the ROM character set.

DIR: Current direction of the ball.

DS: The number of barriers placed in the current round. If DS=0 at the end of a round, then a lucky serve happened.

EN: Controls program flow by determining whether to skip line 300.

N1: The constant 1.

NUM: The actual number of rounds to play.

OXP: Old x position. The x-coordinate of the ball's previous screen position, used in erasing the ball.

OYP: Old y position.

P: Miscellaneous.

PL: Which player is currently playing.

RN: The current round number.

S: Miscellaneous.

SC1: Player 1's score.

SC2: Player 2's score.

SK1: Player 1's skill level.

SK2: Player 2's skill level.

SL(*,*): Stores up to the first 50 barriers placed in each round. SL(*,1) stores the x-coordinate, SL(*,2) stores the y-coordinate, and SL(*,3) stores the type of barrier.

ST(*): Stores the joystick port numbers for the two players.

STR: Score this round. Holds the score earned in the most recently completed round.

T: Used for time delays.

TDIR: Temporary direction. Holds possible new direction of ball after a barrier is hit.

TENS: Used in getting the 10's digit of the number of rounds to play.

TS: Temporary slash (barrier). Since a player can place a barrier only once for every two movements of the ball, TS is set to 1 when a barrier is placed and set to 0 when the ball moves. The player cannot place a barrier when TS equals 1.

TT: Total time used each round. If TT reaches the value of AT, then too much time has gone by.

WUNS: Used in getting the 1's digit of the number of rounds to play.

X: Miscellaneous.

XC(*): Holds the change in the ball's x-coordinate for each of the four directions.

XP: Holds the ball's current x-coordinate.

XT: Holds the x-coordinate of the next location to which the ball might go.

Y: Miscellaneous.

YC(*): Holds the change in the ball's y-coordinate for each of the four directions.

YP: Holds the ball's current y-coordinate.

YT: Holds the y-coordinate of the next position to which the ball might go.

Z: Miscellaneous.

ZZ: Holds the ATASCII value of location on screen during the end-of-round routine.

```
SS SS SS SS SS SS SS SS SS SS SS
SS                             SS
SS         Atari BASIC         SS
SS          'Rebound'          SS
SS Author:  Stephen C. Kuehne 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 #46 SoftSide CV and DV.

Branch vectors.

```
10 GOTO 1000
15 CLR :GOTO 950
```

Set up values.

```
20 SC1=0:SC2=0
25 DIM A$(20):A$=CHR$(165):A$(20)=A$:A
$(2)=A$:N1=1
30 DIM SL(50,3),XC(4),YC(4):XC(1)=0:XC
(2)=1:XC(3)=0:XC(4)=-1:YC(1)=-1:YC(2)=
0:YC(3)=1:YC(4)=0:RN=1
```

Set up values, set colors, and do various routines for player 1.

```
40 AT=1750-150*SK1:SETCOLOR 0,15,8:SET
COLOR 1,0,10:SETCOLOR 2,7,0:SETCOLOR 3
,4,4
45 GOSUB 190:PL=1:GOSUB 150
50 DIR=1:YP=22:XP=INT(18*RND(0))+1:TT=
0:DS=0:STR=0:GOSUB 200:IF DS=0 THEN 50
55 SC1=SC1+STR
```

Set up values, set colors, and do various routines for player 2.

```
60 AT=1750-150*SK2:SETCOLOR 0,15,8:SET
COLOR 1,0,10:SETCOLOR 2,12,0:SETCOLOR
3,4,4
65 GOSUB 190:PL=2:GOSUB 150
70 DIR=1:YP=22:XP=INT(18*RND(0))+1:TT=
0:DS=0:STR=0:GOSUB 200:IF DS=0 THEN 70
75 SC2=SC2+STR
```

Prevent the attract mode, see if the game is over, and return to player 1.

```
80 POKE 77,0:IF RN=NUM THEN 920
85 POSITION 2,0:? #6;"END OF ROUND ";R
N:FOR T=1 TO 500:NEXT T:POSITION 2,0:?
 #6;A$(N1,16);
90 RN=RN+1:GOTO 40
```

Routine to display scores, tell the player to serve, then wait for the serve.

```
150 POSITION 1,0:PRINT #6;"PLAYER 1 SC
ORE=";SC1:FOR T=1 TO 400:NEXT T:POSITI
ON 1,0:PRINT #6;A$(N1,18)
155 POSITION 1,0:PRINT #6;"PLAYER 2 SC
ORE=";SC2:FOR T=1 TO 400:NEXT T:POSITI
ON 1,0:PRINT #6;A$(N1,18)
160 POSITION 4,0:PRINT #6;"PLAYER ";PL
;" UP":FOR T=1 TO 400:NEXT T:POSITION
4,0:PRINT #6;A$(N1,11)
165 POSITION 7,0:PRINT #6;"SERVE":FOR
T=1 TO 400:NEXT T:POSITION 7,0:PRINT #
6;A$(N1,5)
167 IF STRIG(ST(PL))<>0 THEN 167
170 RETURN
```

Routine to draw the border and target.

```
190 POSITION 0,0:PRINT #6;A$
192 FOR X=1 TO 22:POSITION 0,X:PRINT #
6;A$(N1,N1):POSITION 19,X:PRINT #6;A$(
N1,N1):NEXT X
194 POSITION 0,23:PRINT #6;A$;
196 POSITION 9,11:PRINT #6;CHR$(251);C
HR$(253)
198 RETURN
```

Routine to read the joystick. If the player wants to place a slash, make sure it's OK to do so. If so, plot and record the barrier.

```
200 TS=0:P=STICK(ST(PL))
210 IF P>11 THEN 250
220 XT=XP+XC(DIR):YT=YP+YC(DIR)
225 IF XT<1 OR XT>18 OR YT<1 OR YT>22
THEN 250
230 LOCATE XT,YT,Z:POSITION XT,YT
235 IF Z<>32 OR P<9 THEN 240
237 DS=DS+1:TS=1:PRINT #6;"/":IF DS<51
 THEN SL(DS,1)=XT:SL(DS,2)=YT:SL(DS,3)
=1
238 GOTO 250
240 IF Z<>32 THEN 250
242 DS=DS+1:TS=1:PRINT #6;"\":IF DS<51
 THEN SL(DS,1)=XT:SL(DS,2)=YT:SL(DS,3)
=2
```

Routine to figure the new ball position.

```
250 IF EN=1 THEN EN=0:GOTO 302
300 OXP=XP:OYP=YP:GOTO 305
```

Erase the ball, and turn off the sound.

```
302 SOUND 0,0,0,0:POSITION OXP,OYP:PRI
NT #6;" "
```

Increment TT, check whether time is up, and start the time warning beep if necessary.

```
305 TT=TT+1:IF TT>AT THEN 900
306 IF AT-TT<100 THEN SOUND 0,29,10,8
```

Check whether the ball's next position is on the play field.

```
307 XT=XP+XC(DIR):YT=YP+YC(DIR):IF XT>
0 AND XT<19 AND YT>0 AND YT<23 THEN 32
0
```

The ball's next position was off the play field, so reverse its direction.

```
313 DIR=DIR+2:IF DIR>4 THEN DIR=DIR-4
315 SOUND 0,243,10,8:GOTO 300
```

If the ball hit a target, go to the appropriate routine.

```
320 LOCATE XT,YT,Z:IF Z=251 OR Z=253 T
HEN 700
```

If the ball's next position is empty, go to the routine that plots the ball.

```
340 IF Z=32 THEN XP=XT:YP=YT:GOTO 450
350 SOUND 0,121,10,8:XP=XP+XC(DIR):YP=
YP+YC(DIR)
```

Figure the new direction of the ball after it hits a barrier.

```
360 IF ((Z=47 OR Z=40) AND (DIR=2 OR D
IR=4)) OR ((Z=92 OR Z=41) AND (DIR=1 O
R DIR=3)) THEN TDIR=DIR-1
365 IF TDIR=0 THEN TDIR=4
370 IF ((Z=47 OR Z=40) AND (DIR=1 OR D
IR=3)) OR ((Z=92 OR Z=41) AND (DIR=2 O
R DIR=4)) THEN TDIR=DIR+1
375 IF TDIR=5 THEN TDIR=1
```

If the new direction would put the ball off the play field, then don't use it.

```
390 XT=XP+XC(TDIR):YT=YP+YC(TDIR):IF X
T<1 OR XT>18 OR YT<1 OR YT>22 THEN 450
```

Use the new direction, then go back to read the joystick.

```
400 DIR=TDIR:IF TS=1 THEN TS=0:GOTO 30
2
402 EN=1:GOTO 200
```

Clear the sound and the ball's old location, then display the new ball.

```
450 SOUND 0,0,0,0:POSITION OXP,OYP:PRI
NT #6;" "
460 POSITION XP,YP:PRINT #6;CHR$(10)
470 GOTO 200
```

Routine to handle the ball hitting the target.
Clear the sound and the ball's old location.

```
700 SOUND 0,0,0,0:POSITION OXP,OYP:PRI
NT #6;" ":POSITION 9,11:PRINT #6;CHR$(
251);CHR$(253)
```

Sound for a lucky serve.

```
701 IF DS>0 THEN 703
702 FOR X=1 TO 8:SOUND 0,255,10,8:FOR
T=1 TO 15:NEXT T:SOUND 0,0,0,0:FOR T=1
 TO 15:NEXT T:NEXT X:GOTO 705
```

Sound for the ball hitting the target.

```
703 FOR X=50 TO 250 STEP 25:FOR Y=0 TO
 48 STEP 3:SOUND 0,X-Y,10,8:NEXT Y:NEX
T X:SOUND 0,0,0,0:GOTO 720
```

Tell the player to serve again, then wait for the serve.

```
705 POSITION 4,0:PRINT #6;"LUCKY SERVE
":FOR T=1 TO 400:NEXT T:POSITION 4,0:P
RINT #6;A$(N1,11)
710 POSITION 4,0:PRINT #6;"SERVE AGAIN
":FOR T=1 TO 400:NEXT T:POSITION 4,0:P
RINT #6;A$(N1,11)
712 IF STRIG(ST(PL))<>0 THEN 712
715 RETURN
```

Highlight the slashes most recently placed, and figure the score.

```
720 IF DS>50 THEN DS=50
721 FOR Z=1 TO DS:X=SL(Z,1):Y=SL(Z,2):
LOCATE X,Y,ZZ:IF SL(Z,3)<>1 THEN 745
725 FOR S=15 TO 0 STEP -1:POSITION X,Y
:PRINT #6;"%":POSITION X,Y:PRINT #6;CH
R$(ZZ):SOUND 0,60,8,S:NEXT S
```

```
730 GOTO 760
745 FOR S=15 TO 0 STEP -1:POSITION X,Y
:PRINT #6;"%":POSITION X,Y:PRINT #6;CH
R$(ZZ):SOUND 0,60,8,S:NEXT S
760 NEXT Z
770 IF DS<16 THEN STR=16-DS
780 RETURN
```

Time ran out, so give a bonus to the player's opponent.

```
900 POSITION OXP,OYP:? #6;" ":SOUND 0,
0,0,0
903 POSITION 4,0:PRINT #6;"TIME'S UP";
:FOR T=1 TO 400:NEXT T:POSITION 4,0:?
#6;A$(N1,9)
910 IF PL=2 THEN 914
911 SC2=SC2+25:POSITION 3,0:? #6;"PLAY
ER 2 GETS";:FOR T=1 TO 300:NEXT T:POSI
TION 3,0:? #6;A$(N1,13)
912 GOTO 915
914 SC1=SC1+25:POSITION 3,0:? #6;"PLAY
ER 1 GETS";:FOR T=1 TO 300:NEXT T:POSI
TION 3,0:? #6;A$(N1,13)
915 POSITION 2,0:? #6;"25 BONUS POINTS
";:FOR T=1 TO 300:NEXT T:POSITION 2,0:
? #6;A$(N1,15)
917 POP :ON PL GOTO 60,80
```

Game over.

```
920 ? #6;CHR$(125):PRINT #6:PRINT #6:P
RINT #6;"FINAL SCORE:"
925 PRINT #6:PRINT #6;"PLAYER 1 - ";SC
1
927 PRINT #6;"PLAYER 2 - ";SC2
930 POSITION 0,15:PRINT #6;"PUSH TRIGG
ER TO":PRINT #6;"PLAY AGAIN"
935 IF STRIG(0)<>0 AND STRIG(1)<>0 THE
N 935
940 PRINT #6;CHR$(125):FOR T=1 TO 50:N
EXT T:GOTO 15
```

Determine the number of joysticks available.

```
950 DIM ST(2):ST(1)=0:PRINT #6;CHR$(12
5):POSITION 0,0:PRINT #6;"ARE BOTH PLA
YERS US-";
951 ? #6;"ING THE SAME JOY-   STICK?":
ST(2)=0
952 POSITION 8,4:? #6;" YES":POSITION
8,5:? #6;" NO"
953 SOUND 0,121*(ST(2)+1),10,8:POSITIO
N 8,4+ST(2):PRINT #6;">":IF STRIG(0)=0
 THEN SOUND 0,0,0,0:GOTO 960
954 IF STICK(0)=13 AND ST(2)=0 THEN ST
(2)=1:GOTO 952
956 IF STICK(0)=14 AND ST(2)=1 THEN ST
(2)=0:GOTO 952
958 GOTO 953
```

Get player 1's skill level.

```
960 FOR T=1 TO 35:NEXT T:PRINT #6;CHR$
(125):POSITION 0,0
962 PRINT #6;"SKILL PLAYER 1":FOR X=1
TO 10:POSITION 5,X+5:PRINT #6;X:NEXT X
:SK1=1
964 SOUND 0,100+SK1*15,10,8:POSITION 4
,SK1+5:PRINT #6;">":IF STRIG(0)=0 THEN
 980
966 IF STICK(0)=14 AND SK1>1 THEN POSI
TION 4,SK1+5:PRINT #6;" ":SK1=SK1-1:GO
TO 964
968 IF STICK(0)=13 AND SK1<10 THEN POS
ITION 4,SK1+5:PRINT #6;" ":SK1=SK1+1:G
OTO 964
970 GOTO 964
```

Get player 2's skill level.

```
980 FOR T=1 TO 35:NEXT T:PRINT #6;CHR$
(125):POSITION 0,0
981 PRINT #6;"SKILL PLAYER 2":FOR X=1
TO 10:POSITION 15,X+5:PRINT #6;X:NEXT
X:SK2=1
982 SOUND 0,100+SK2*15,10,8:POSITION 1
4,SK2+5:PRINT #6;">":IF STRIG(ST(2))=0
 THEN 987
983 IF STICK(ST(2))=14 AND SK2>1 THEN
POSITION 14,SK2+5:PRINT #6;" ":SK2=SK2
-1
984 IF STICK(ST(2))=13 AND SK2<10 THEN
 POSITION 14,SK2+5:PRINT #6;" ":SK2=SK
2+1
985 GOTO 982
```

Get the number of rounds to play.

```
987 FOR T=1 TO 35:NEXT T:? #6;CHR$(125
):POSITION 0,0
988 ? #6;"HOW MANY ROUNDS DO  YOU WANT
 TO PLAY?":FOR X=0 TO 9:POSITION 8,X+6
:? #6;X;" ";X:NEXT X:TENS=0
```

```
989 SOUND 0,115+TENS*15,10,8:POSITION
7,TENS+6:? #6;">":IF STRIG(0)=0 THEN 9
94
990 IF STICK(0)=14 AND TENS>0 THEN POS
ITION 7,TENS+6:? #6;" ":TENS=TENS-1
991 IF STICK(0)=13 AND TENS<9 THEN POS
ITION 7,TENS+6:? #6;" ":TENS=TENS+1
992 GOTO 989
994 FOR T=1 TO 35:NEXT T:WUNS=0:IF TEN
S=0 THEN WUNS=1:POSITION 10,6:? #6;" "
995 SOUND 0,115+WUNS*15,10,8:POSITION
11,WUNS+6:? #6;"<":IF STRIG(0)=0 THEN
999
996 IF STICK(0)=14 AND ((TENS=0 AND WU
NS>1) OR (TENS>0 AND WUNS>0)) THEN POS
ITION 11,WUNS+6:? #6;" ":WUNS=WUNS-1
997 IF STICK(0)=13 AND WUNS<9 THEN POS
ITION 11,WUNS+6:? #6;" ":WUNS=WUNS+1
998 GOTO 995
999 NUM=TENS*10+WUNS:SOUND 0,0,0,0:PRI
NT #6;CHR$(125):GOTO 20
```

Transfer the character set from ROM to RAM, then redefine eight characters.

```
1000 CHBAS=256*(PEEK(106)-4):POKE 106,
PEEK(106)-5:GRAPHICS 17:CHROM=256*PEEK
(756)
1005 POSITION 6,5:PRINT #6;"rebound":P
OSITION 1,8:PRINT #6;"BY stephen kuehn
e"
1007 POSITION 3,15:PRINT #6;"TRANSFERR
ING":POSITION 2,16:PRINT #6;"CHARACTER
 SET."
1010 POSITION 3,17:? #6;"PLEASE WAIT."
:FOR X=0 TO 1023:POKE CHBAS+X,PEEK(CHR
OM+X):NEXT X
1020 POKE 756,CHBAS/256
1030 FOR X=1 TO 8:READ P:FOR Y=0 TO 7:
READ Z:POKE CHBAS+P*8+Y,Z:NEXT Y:NEXT
X
1100 DATA 60,0,64,32,16,8,4,2,0
1110 DATA 15,0,2,4,8,16,32,64,0
1120 DATA 10,60,126,255,255,255,255,12
6,60
1130 DATA 5,255,255,255,255,255,255,25
5,255
1140 DATA 59,127,255,255,255,255,255,2
55,127
1150 DATA 61,254,255,255,255,255,255,2
55,254
1160 DATA 8,0,2,4,8,16,32,64,0
1170 DATA 9,0,64,32,16,8,4,2,0
```

Display the instructions.

```
1200 PRINT #6;CHR$(125):POSITION 6,0:P
RINT #6;"rebound":PRINT #6
1210 PRINT #6;"THE OBJECT OF RE-":PRIN
T #6;"BOUND IS TO HIT THE":PRINT #6;"T
ARGET WITH THE BALL";
1220 PRINT #6;"USING AS FEW SLASHES";:
PRINT #6;"AS POSSIBLE.  YOU":PRINT #6;
"MUST USE AT LEAST 1"
1230 PRINT #6;"SLASH, AND EACH EX-":PR
INT #6;"TRA SLASH AFTER THE":PRINT #6;
"1ST DECREASES YOUR"
1240 PRINT #6;"POSSIBLE SCORE FOR":PRI
NT #6;"THAT ROUND.  IF YOU":PRINT #6;"
USE ONLY 1 SLASH,"
1250 PRINT #6;"YOU WILL RECEIVE 15":PR
INT #6;"POINTS.  IF YOU USE":PRINT #6;
"2 SLASHES, YOU WILL"
1260 PRINT #6;"RECEIVE 14 POINTS,":PRI
NT #6;"AND SO ON, SO THAT":PRINT #6;"I
F YOU USE MORE THAN";
1270 PRINT #6;"15 SLASHES, YOU WILL";:
PRINT #6;"RECEIVE NO POINTS."
1273 ? #6:? #6;"   PRESS TRIGGER.";
1275 IF STRIG(0)<>0 AND STRIG(1)<>0 TH
EN 1275
1276 PRINT #6;CHR$(125):FOR T=1 TO 50:
NEXT T:POSITION 0,0
1277 PRINT #6;"YOU HAVE A LIMITED":PRI
NT #6;"AMOUNT OF TIME TO":PRINT #6;"HI
T THE TARGET.  A"
1278 PRINT #6;"FAST HIGH BEEP WARNS";:
PRINT #6;"YOU WHEN YOUR TIME":PRINT #6
;"IS ALMOST UP.":PRINT #6
1280 PRINT #6;"THE TRIGGER SERVES.":PR
INT #6;"PUSHING THE JOYSTICK";:PRINT #
6;"LEFT PUTS A / IN"
1290 PRINT #6;"FRONT OF THE BALL.":PRI
NT #6;"PUSHING THE JOYSTICK";:PRINT #6
;"RIGHT PUTS A \ IN"
1300 PRINT #6;"FRONT OF THE BALL."
1310 PRINT #6:PRINT #6;"PRESS TRIGGER
TO":PRINT #6;"BEGIN."
```

```
1320 IF STRIG(0)<>0 AND STRIG(1)<>0 TH
EN 1320
1330 PRINT #6;CHR$(125):FOR T=1 TO 50:
NEXT T
1340 GOTO 15
```

STOMP TABLE

For **Atari® REBOUND**

LINES	STOMP CODE	LENGTH
10 - 70	ML	534
75 - 167	EU	514
170 - 235	KF	350
237 - 315	JW	448
320 - 460	KL	515
470 - 720	NW	509
721 - 911	JO	523
912 - 950	UM	544
951 - 966	BW	554
968 - 988	QO	577
989 - 997	RR	513
998 - 1110	ZB	519
1120 - 1230	GP	558
1240 - 1277	RM	550
1278 - 1340	GX	454

General Information

These are the standard procedures for the programs published by **SoftSide** Publications, Inc. 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.

STOMP Tables

At the conclusion of each **SoftSide** listing, we include a **STOMP** Table. **STOMP** for the Atari appeared in Issue #45. **STOMP** supersedes **SWAT** as **SoftSide's** standard debugging tool to help those who type BASIC programs from the pages of **SoftSide Selections.** If you don't have **STOMP,** we'll send you a free reprint. Send a business-sized, self-addressed, stamped envelope to:

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

Be sure to tell us what kind of computer you have.

Line Listings

IBM®PC USERS

The line listings in this booklet appear in a 40-column format. If you type LIST when your computer is displaying 40-column text (WIDTH 40), your screen should display exactly what appears in the printed listing. Be sure to use **STOMP** on your program, and get the free reprint if you don't have **STOMP.**

APPLE® USERS

The line listings in this booklet are in standard Applesoft® format, and they appear exactly as they should on your screen when you type LIST. Beginning with Issue #45, Applesoft listings in **SoftSide Selections** may have lower-case characters in them. If you have an Apple IIe or an Apple II with a lower-case adapter, you may type these listings exactly as they appear. If you have an Apple II without lower case, simply type the lower-case characters as capitals. **STOMP, SoftSide's** debugging utility, ignores the case of characters, so both ways of typing programs result in the same **STOMP** Table.

Things to watch out for when typing are:

- Lower-case characters: The Apple IIe and some Apple II's have lower case. Many programs we publish contain lower case. Since, **STOMP, SoftSide's** debugging utility, ignores the case of characters, you may use capital letters if you so desire or if your Apple doesn't have lower case.

- REM and DATA statements: **STOMP,** like Applesoft, ignores REM statements. You do not have to type the text of REMs. **STOMP** also ignores the space or spaces between the keyword DATA and the first data element. Type numbers and strings in DATA statements exactly as they appear in the published listing.
- Spaces between quotes: Applesoft is a bit eccentric about how it shows these. Just list the line after you type it, and compare it to the printed listing.

Also, be sure to use **STOMP** on your program, and get the free reprint if you don't have **STOMP.**

ATARI® USERS

- The line listings in this booklet 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 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 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.

The Best of SoftSide

NOW AVAILABLE ON DISK

☐ **YES...**I want to enjoy the best programs — Games, Utilities, Adventures — from SoftSide Magazine's past. Please rush my copies of The Best of SoftSide to:

Name ________________________________

Address ________________________________

City ____________________ State ________ Zip ________

Here's my order For The Best of SoftSide

Computer	Quantity Book Only $14.95 ea.	Quantity Book plus Disk $49.95 ea.
Apple®		
Atari®		
TRS-80®		
TOTAL COPIES	@$14.95	@$49.95

☐ I've already ordered The Best of SoftSide book. I'm enclosing $35 for the disks only.

I'm paying by:

☐ Check # ________________

☐ Money Order # ________________

☐ VISA

☐ MasterCard

Total Enclosed $________________

(Foreign order please include $5 postage PAY IN USA FUNDS ONLY.)

VISA/MasterCard # ________________________ Exp. Date ________

Name of Cardholder ________________________

Signature ________________________

Like SoftSide? Wait 'til you've seen our BEST!

The Best of SoftSide!

For the past four years, **SoftSide** Magazine has been bringing Apple®, Atari®, and TRS-80® owners the best in BASIC software. But now you can do even better...**The Best of SoftSide.** From all our back issues, we've selected the most useful...the most entertaining...the most fun programs **SoftSide** has ever published.

The Best of SoftSide is available in three versions...one for Apple, Atari, and TRS-80. Each contains page after page of BASIC code for adventures, simulations, practical applications, and much more.

To order your copy of **The Best of SoftSide,** fill out the coupon, and mail it along with $14.95 to **SoftSide,** 10 Northern Blvd., Northwood Executive Park, Amherst, New Hampshire 03031.

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

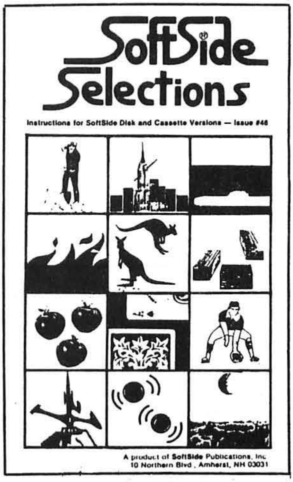
ORDER THIS ISSUE'S MEDIA

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.

☐ **#46 DV $19.95** ☐ **#46 CV $14.95**

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 #/VISA # __________________________
Exp. Date____/____/____ Signature__________________



SoftSide® Selections

Instructions for SoftSide Disk and Cassette Versions — Issue #46

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

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

SoftSide®

FRONT RUNNER

CRISIS

A ruthless race of aliens has taken possession of the Empire State Building. Your goal is to rescue the hostages.

KANGARILLA

Your baby kangaroo has wandered to the top of a four-level maze, and you must rescue him.

REBOUND

Fire the moving ball into the colored goal in the center of your screen, but count the barriers you erect with each press of the joystick. They block or bounce your ball in the right direction — and lower your score as they accumulate.

SOFTSIDE PUBLICATIONS, INC.
Amherst, New Hampshire