

FILE = PA
BLK= 0
01 (GAME PARAMETERS (111))
114 C= PRDM (PLAYER-REVEALER-HOSTAGE MAX TIME BASE)
2118 C= H-H-D (HOSTAGE TO PLAYER DISTANCE)
3112 C= H-H-D (HOSTAGE TO HOSTAGE DISTANCE)
4178 C= ONTOPLMT (WITHIN SPACING)
5488 C= MAXASOM (MAX ASSIMILATION DISTANCE)
61
717 C= HCOLS 5 C= HROWS
81
SINGLES HROWS * C= HROWL2
91
101-->
111
121
131
141
151

roto
11/20

PUSH:CCRD ?

FILE = RA

BLK= 0

0! GAME PARAMETERS !!!!)

1!4 C= PRTBM (PLAYER-REVEALER-HOSTAGE MAX TIME BASE)

2!18 C= H-P-D (HOSTAGE TO PLAYER DISTANCE)

3!12 C= H-H-D (HOSTAGE TO HOSTAGE DISTANCE)

4!18 C= ONTOPLMT (WITHIN SPACING)

5!88 C= MAXASSM (MAX ASSIMILATION DISTANCE)

6!

7!7 C= NCOLS 5 C= NROWS

8! NCOLS NROWS * C= NNODES

9!

10! -->

11!

12!

13!

14!

15!

FILE = TST

BLK= 0

```
0:( TEST SHIT TO DUMP NEAT OVECTOR STUFF )
1:HEX V= VERBADR
2:SUBR NEWINTR B LDAX, B INX, A L MOV, B LDAX, B INX, A H MOV,
3:VERBADR SHLD, PCHL,
4:CODE ZAMMER NEWINTR Y LXIX, NEXT
5: LSAT ZAMMER VERBADR H. VERBADR 1+ SPACE H. ;
6:DECIMAL ;S
7: VD CR ." C= " DUP NOWC OVBe .
8:." R= " DUP NOWR OVBe .
9:." D= " DUP NOWD OVBe .
10:." DIS= " DUP DISTANCE OV@ H.
11:." DD= " DUP DELTADJST OV@ H.
12:." MAXD= " DUP MAXDIST OVBe H.
13:." X= " DUP VX OV@ H.
14:." Y= " VY OV@ H. CR ;
15:-->
```

FILE = VE

BLK= 0

0:(CUSTOM VECTOR FIELDS)
1:DECIMAL VLENGTH SC= INTR NC= INTC (INITIAL POS AND COL)
2:INC= NOWR NC= NOWC (CURRENT ROW AND COLUMN)
3:INC= NOWD (CURRENT DIRECTION)
4:(NC= NXTR NC= NXTC) (NEXT ROW AND COLUMN)
5:(CUSTOM VECTOR ROUTINE GOODIES)
6:INC= BASEX 1+
7:INC= BASEY 1+
8:INC= DELTAX 1+
9:INC= DELTAY 1+
10:INC= MAXDIST MAXDIST NOWR - 1+ C= POSLEN
11:INC= DISTANCE 1+
12:INC= DELTADIST 1+ DELTADIST NOWR - C= SNATLEN
13:(NC= ACCDIST 1+)
14:INC= MEMDIST NC= MEMR NC= MEMC NC= MEMD
15:-->

BLK= 1

0:(MORE CUSTOM VECTOR FIELDS)
1:INC= CUSVEC 1+ (CUSTOM VECTOR ROUTINE ADDRESS)
2:INC= MYTYPE (VECTOR TYPE INDICATOR)
3:INC= MYFLAG (BUILD IN NEATO FLAG) NC= FLAGCODE
4:INC= MYFACE 1+ (WHAT I LOOK LIKE IN THE OPEN)
5:INC= VCOR 1+ (MY COROUTINE CELL)
6:INC= BEHIND 1+ (FELLOW BEHIND ME)
7:INC= AHEAD 1+ (FELLOW AHEAD OF ME)
8:INC= VISFLAG
9:1+ C= MLENGTH
10:MLENGTH SC= HOSSV NC= ASSMSV
11:INC= VIRGIN
12:INC= DIST-1 (PREV DISTANCE)
13:INC= DISPF (DISPLACEMENT FACTOR)
14:INC= SNATCHER 1+
15:1+ C= HLENGTH -->

BLK= 2

0:(MORE UNIQUE VECTOR STUFF)
1:MLENGTH AHEAD C= MYSLAVE
2:SC= FNDPTR FNDPTR C= TRACKPTR 1+ NC= TREECK 1+
3:BEHIND C= MYBOSS
4:INC= FRONTIER 1+
5:INC= VISMAT NCOLS + C= TREES
6:0 SC= TPL NC= TPH NC= TC NC= TR NC= TD 1+ C= TEL
7:TEL NNODES * C= TDEPTH 70 C= SURPLUS
8:TDEPTH TREES + SURPLUS +
9:1+ C= MONLEN
10:HLENGTH C= PLENGTH (PLAYERS VECTOR LENGTH)
11:PLENGTH C= RVLENGTH (REVEALERS LENGTH)
12:(BITS AND CODES)
13:(VECTOR TYPES)
14:0 SC= T-TYP NC= H-TYP NC= M-TYP C= K-TYP
15:-->

FILE = VE

BLK= 3

```
0:( HOSTAGE AND PLAYER STATE VARIABLES )
1:( THE HOSTAGE STATE VARIABLE )
2:0 SC= HSFREE ( HOSTAGE FREE )
3:1NC= HSATP ( HOSTAGE ATTACHED TO PLAYER )
4:1NC= HSATM DROP ( HOSTAGE ATTACHED TO MONSTER )
5:( ASSIMILATION STATE VARIABLE )
6:0 SC= ASNOT ( NOT ASSIMILATED )
7:1NC= ASSIM DROP ( FULLY ASSIMILATED )
8:( PLAYERS ASSIMILATION STATE VARIABLE )
9:0 SC= ASCOOL ( PLAYER IS SPIFFY )
10:1NC= ASONTOP DROP ( PLAYER IS ON TOP OF HOSTAGES )
11!-->
12!
13!
14!
15!
```

BLK= 4

```
0:( VECTOR STUFF ) VPTR @ HEX FFF0 VPTR ! <STKD
1:RAMMARK MLENGTH BR= BKGV RAMLEN C= BKVL VARHERE C= BKVS
2:RAMMARK PLENGTH BR= PLYRV RAMLEN C= PLYRL VARHERE C= PLYVS
3:RAMMARK RVLENGTH BR= REVV RAMLEN C= REVL VARHERE C= REVS
4:RAMMARK MLENGTH BR= TV1 RAMLEN C= TVVL VARHERE C= TVVS
5:RAMMARK RVLENGTH BR= RVOV
6:RVLENGTH BR= RV1V
7:RVLENGTH BR= RV2V
8:RVLENGTH BR= RV3V
9:RVLENGTH BR= RV4V
10:RVLENGTH BR= RV5V
11:RVLENGTH BR= RV6V
12:RVLENGTH BR= RV7V RAMLEN C= RVVL VARHERE C= RVVS
13:STK> VPTR @ H. VPTR !
14!-->
15!
```

BLK= 5

```
0:( MONSTER STUFF )
1!
2:RAMMARK MONLEN BR= MONV1
3:MONLEN BR= MONV2
4:MONLEN BR= MONV3
5:MONLEN BR= MONV4
6:RAMLEN C= MONVL VARHERE C= MONVS
7:MONLEN C= MONVBYTES
8:DECIMAL -->
9!
10!
11!
12!
13!
14!
15!
```

FILE = VE

BLK= 6

```
01( TREASURE VECTORS )
11
21RAMMARK MLENGTH BR= TRSV1
31MLENGTH BR= TRSV2 MLENGTH BR= TRSV3
41MLENGTH BR= TRSV4
51RAMLEN C= TRSVL VARHERE C= TRSVS
61MLENGTH C= TRSVBYTES
714 C= TOTAL-JEWELS
81-->
91
101
111
121
131
141
151
```

BLK= 7

```
01( HOSTAGE VECTORS )
11RAMMARK HLENGTH BR= HOSV1
21HLENGTH BR= HOSV2 HLENGTH BR= HOSV3
31HLENGTH BR= HOSV4
41RAMLEN C= HOSVL VARHERE C= HOSVS
51HLENGTH C= HOSVBYTES
614 C= TOTAL-HOSTAGES
71TABLE HOSTAB HOSV1 , HOSV2 , HOSV3 , HOSV4 , 0 ,
81
91( ***** )
101HOSV4 C= TEMRM
111-->
121
131
141
151
```

BLK= 8

```
01( MORE NEAT VECTOR STUFF )
11: ZAP:VECT 0 RV7V BKGV RV7V - BKVL + FILL
210 HOSV4 MONV1 HOSV4 - MLENGTH + FILL ;
31-->
41
51
61
71
81
91
101
111
121
131
141
151
```

FILE = VE

BLK= 9

```
0:( SPECIAL VECTOR GETTERS AND PUTTERS )
1:CODE PUSH:CCR O H MVI, H D MOV, Y PUSHX, vaddr LIYD,
2:NOWC Y L LDX, NOWR Y E LDX, Y POPX, H PUSH, D PUSH, NEXT
3:
4:CODE COGO ( exchase BC with VCOR )
5: vaddr LHLD, VCOR D LXI, D DAD,
6: M A MOV, C M MOV, A C MOV, H INX,
7: M A MOV, B M MOV, A B MOV, NEXT
8: SETCO 1+ VCOR V! ;
9:
10:
11:-->
12:
13:
14:
15:
```

FILE = VA

BLK= 0

```
01( GAME CONTROL PARAMETERS )
11BV= NOBREAK
21V= TRASHFLAG.
31V= GAME-OVER V= GAME#
41V= NPLAYERS V= PLAYERUP
51V= INITIAL-LIVES
61V= REMAINING-LIVES
71V= PLAYERDEAD ( PLAYER NAILED BY MONSTER FLAG )
81V= PLAYERVELO ( PLAYER VELOCITY )
91BV= FREEZEFLAG ( PLAYER MOTION FREEZE FLAG )
101BV= SMARTS ( MONSTER SMARTNESS FACTOR )
111V= MONSTERCOUNT ( # OF MONSTERS FLOATING AROUND )
121BV= BANC BV= BANR ( POINT OF BANISHMENT FOR MONSTER )
131BV= IBNC BV= IBNR ( POINT OF INITIAL RETURN FOR MONSTER )
141-->
151
```

BLK= 1

```
01( MORE VARIABLES )
11V= TOTAL-CONNECTS V= OLD-CONNECTS
21V= TOTAL-REVEALED-GROTTOS
31V= KEY-THRESHOLD
41V= TOTAL-PATHS
51V= REVEALED-PATHS ( # OF PATHS REVEALED TO PLAYER SO FAR )
61V= START-COL V= STOP-COL
71V= FOUNDIT BV= THATSALL
81DECIMAL -->
91
101
111
121
131
141
151
```

BLK= 2

```
01( FREEZE AND UNFREEZE ROUTINES )
11SUBR FREEZE FREEZEFLAG H LXI, M INR, RET,
21SUBR FREEZE? FREEZEFLAG LDA, A ANA, RET,
31CODE FREEZETH FREEZE CALL, NEXT
41CODE UNFREEZE FREEZEFLAG H LXI, M DCR, 0<, IF, 0 M MVI, THEN,
51NEXT
61-->
71
81
91
101
111
121
131
141
151
```


FILE = D1

BLK= 0

```
0:( NEW SQUARE ROOT ROUTINE )
1:IF= sqrt1
2:SUBR sqrt <ASSEMBLE
3:1 A MVI, 1 B LXI, 1 D LXI,
4:LABEL sqrt1 A ANA, D DSBC, RZ, RC, D DAD, B INX, B JNX,
5:XCHG, B DAD, A INR, XCHG, sqrt1 JMPR, ASSEMBLE>
6:-->
7:
8:
9:
10:
11:
12:
13:
14:
15:
```

BLK= 1

```
0:( 16 BIT INTEGER DIVIDE ROUTINE: M N UN/ Q R ) DECIMAL
1:FORWARD .ZERO FORWARD IDV50 FORWARD IDV60
2:FORWARD IDV10 FORWARD IDV20 FORWARD IDV30 FORWARD IDV40
3:SUBR unsddiv <ASSEMBLE L C MOV, H B MOV, D A MOV, O H LXI,
4:E ORA, .ZERO JRZ, B A MOV, 16 B MVI,
5:LABEL IDV10 C RALR, RAL, H DADC, D DSBC,
6:LABEL IDV20 CMC, IDV50 JRNC,
7:LABEL IDV30 IDV10 DJNZ, IDV60 JMPR,
8:LABEL IDV40 C RALR, RAL, H DADC, A ANA, D DADC,
9:IDV30 JRC, IDV20 JRZ,
10:LABEL IDV50 IDV40 DJNZ, D DAD, A ANA, ( MAKE IT POS )
11:LABEL IDV60 C RALR, RAL, A D MOV, C E MOV,
12:LABEL .ZERO RET, ASSEMBLE>
13:SUBR UNSDIV H PUSH, D DSBC, CY, IF, O D LXI, H POP, ELSE,
14:H POP, unsddiv CALL, THEN, RET, CODE UN/ EXX, D POP, H POP,
15:UNSDIV CALL, H PUSH, D PUSH, EXX, NEXT DECIMAL -->
```


BLK= 2

```
0:( COMPUTE DELTA FOR 1 COORDINATE - CLEAR VECTOR )
1:( FIRST A NEGATION SUBROUTINE )
2:SUBR CMPHL H A MOV, CMA, A H MOV, L A MOV, CMA, A L MOV, H INX,
3:RET,
4:( IN: HL=TARGET, DE=TIME, BC=START )
5:SUBR CDELTA B PUSH, A ANA, B DSBC, CY~, IF, UNSDIV CALL,
6:ELSE, CMPHL CALL, UNSDIV CALL, CMPHL CALL, XCHG, CMPHL CALL,
7:XCHG, THEN, B POP, B DAD, RET,
8:DECIMAL -->
9:
10:
11:
12:
13:
14:
15:
```

FILE = DI

BLK= 3

0!(ROUTINE TO VECTOR BETWEEN CURRENT POSITION AND DEST
1!IN TIME GIVEN IN VECTOR) -->
2!CODE A->DEST/TIME B PUSH, Y PUSHX,
3!vaddr LIYD,
4!VXH Y B LDX, VX Y C LDX, VDESTXH Y H LDX, VDESTX Y L LDX,
5!TTIMERH Y D LDX, TTIMER Y E LDX, D PUSH, CDELTA CALL,
6!H VXH Y STX, L VX Y STX, D VDXH Y STX, E VDX Y STX,
7!VYH Y B LDX, VY Y C LDX, VDESTYH Y H LDX, VDESTY Y L LDX,
8!D POP, CDELTA CALL,
9!H VYH Y STX, L VY Y STX, D VDYH Y STX, E VDY Y STX,
10!Y POPX, B POP, NEXT
11!DECIMAL -->
12!
13!
14!
15!



FILE = NM

BLK= 3

```
0!( TEST:REL AND MOVE:NODE )
1!( D=ROW,E=COL,C=REL COL ROW REL TEST:REL --- DIST )
2!SUBR test:rel C A MOV, MPLO ADI, A C MOV, node^ CALL,
3!M A MOV, RET,
4!CODE TEST:REL EXX, B POP, H POP, D POP, L D MOV, test:rel CALL,
5!A L MOV, O H MVI, H PUSH, EXX, NEXT
6!( MOVE:NODE TABLES )
7!DATA xtbl -1 B, 0 B, 1 B, -1 B, 1 B, -1 B, 0 B, 1 B,
8!DATA ytbl 1 B, 1 B, 1 B, 0 B, 0 B, -1 B, -1 B, -1 B,
9!SUBR move:node B PUSH, ( C=DIR, D=ROW,E=COL )
10!O B MVI, ytbl H LXI, B DAD, M A MOV, D ADD, A D MOV,
11!xtbl H LXI, B DAD, M A MOV, E ADD, A E MOV, B POP, RET,
12!CODE MOVE:NODE EXX, B POP, H POP, D POP, L D MOV,
13!move:node CALL, D L MOV, O D MVI, D H MOV,
14!D PUSH, H PUSH, EXX, NEXT
15!-->
```

BLK= 4

```
0!( STUFF )
1!: NODE! NODE^ ! ;
2!: NODE@ NODE^ @ ;
3!: NODEB@ NODE^ B@ ;
4!: CLEAR:NODEMAT 0 0 NODEMAT NODEMAT:SIZE FILL ;
5!-->
6!
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

1011
E3



BLK= 5

```
0!( CHECK:NODE AND ESTVALDIR )
1!HEX
2!: CHECK:NODE 2DUP 7F AND NROWS <
3!SWAP 7F AND NCOLS < AND ;
4!DECIMAL
5!F= EVDL
6!SUBR estvaldir <ASSEMBLE NOWR Y D LDX, NOWC Y E LDX,
7!O NOWD Y MVIX,
8!LABEL EVDL NOWD Y A LDX, MPLO ADI, A C MOV, node^ CALL,
9!M A MOV, A ANA, RNZ, NOWD Y INRX, EVDL JMPR, ASSEMBLE>
10!CODE ESTVALDIR B PUSH, Y PUSHX, vaddr LIYD, estvaldir CALL,
11!Y POPX, B POP, NEXT
12!
13!-->
14!
15!
```

FILE = NM

BLK= 6

0! (NODE MATRIX MANIPULATORS
1! V= NR:C V= NR:R V= NR:M
2! GET:C:R:M NR:C @ NR:R @ NR:M @ ;
3! SET:DRAWN ROLL DRAWMSK NODE^ SET ;
4! TEST:DRAWN ROLL DRAWMSK NODE^ BIT ;
5! SET:GROTTO:DRAWN DRAWFLG NODE^ BONE ;
6! TEST:GROTTO:DRAWN DRAWFLG NODEB@ ;
7! -->
8!
9!
10!
11!
12!
13!
14!
15!

FILE = NM

BLK= 0

```
0!( MESH PARAMETERS ) <STKD
1!
2!320 NCOLS / C= COLSIZE 180 NROWS / C= ROWSIZE
3!36 C= COLGUARD 28 C= ROWGUARD 10 C= CIR-RAD
4!8 C= HOLE-RAD NROWS 1- C= START-ROW
5!COLSIZE COLGUARD - C= COLDEV ROWSIZE ROWGUARD - C= ROWDEV
6!
7! COLCENT COLSIZE * COLSIZE 2 / + 160 - ;
8! ROWCENT ROWSIZE * ROWSIZE 2 / + 104 - ;
9!
10! COMP:X COLCENT COLDEV 2 / COLDEV RND - + ;
11! COMP:Y ROWCENT ROWDEV 2 / COLDEV RND - + ;
12!
13! COMP:XY COMP:Y SWAP COMP:X SWAP ;
14!STK> -->
15!
```

BLK= 1

```
0!( MESH MATRIX GOODIES )
1!0 SC= NODX NC= NODXH
2!NC= NODY NC= NODYH NC= NBX 1+ NC= NBY 1+
3!NC= MPL0 NC= MPL1 NC= MPL2 NC= MPL3 NC= MPL4 NC= MPL5
4!NC= MPL6 NC= MPL7
5!NC= NDX0 NC= NDX1 NC= NDX2 NC= NDX3 NC= NDX4 NC= NDX5
6!NC= NDX6 NC= NDX7
7!NC= NDY0 NC= NDY1 NC= NDY2 NC= NDY3 NC= NDY4 NC= NDY5
8!NC= NDY6 NC= NDY7
9!NC= CONFLG NC= #CON
10!NC= DRAWFLG NC= DRAWMSK
11!NC= >TREASURE 1+
12!1+ C= NODSIZ
13!NODSIZ NNODES * C= NODEMAT:SIZE
14!NODEMAT:SIZE BA= NODEMAT -->
15!
```

BLK= 2

```
0!( NODE ZAMMERS )
1!( SUBR node^ D= ROW E= COL C= DISP, OUT HL= ^ )
2!F= N^1 F= N^2 SUBR node^ <ASSEMBLE D PUSH, B PUSH,
3!D B MOV, B INR, NCOLS MINUS A MVI,
4!LABEL N^1 NCOLS ADI, N^1 DJNZ, E ADD, A INR, A B MOV,
5!NODSIZ MINUS H LXI, NODSIZ D LXI,
6!LABEL N^2 D DAD, N^2 DJNZ, B DAD, 0 NODEMAT B LXI, B DAD,
7!B POP, D POP, RET, ASSEMBLE>
8!CODE NODE^ EXX, B POP, H POP, D POP, L D MOV, node^ CALL,
9!H PUSH, EXX, NEXT
10!-->
11!
12!
13!
14!
15!
```

FILE = VC

BLK= 0

```
0!( STUFF )
1! RETURN:INITIAL:POSITION INTR VB@ NOWR VB! INTC VB@ NOWC VB! ;
2!
3!-->
4! CRAWL:TUNNEL
5!4 NDUP DROP TEST:REL SWAP *
6!TIMER!--ON MOVE:NOD
7!2DUP NODX NODE@ DESTX! NODY NODE@ DESTY!
8!A->DEST/TIME ;
9! SET:POSITION:XY PUSH:CCR 2DUP NODX NODE@ X! NODY NODE@ Y! ;
10! RETURN:INITIAL:POSITION INTR VB@ NOWR VB! INTC VB@ NOWC VB! ;
11!-->
12!
13!
14!
15!
```

BLK= 1

```
0!( MORE STUFF )
1! SET:NEW:MCCR NOWR VB! NOWC VB! ;
2! SET:INITIAL:MCCR DUP ROLL INTR OVB! INTC OVB! ;
3!( RUSH TO DESTINATION )
4! ON:TARGET? PUSH:CCR INTR VB@ = SWAP INTC VB@ = AND ;
5!-->
6! RUSSIAN STAY:PUT PUSH:CCR NODX NODE@ DESTX! ZEROTIMEB
7!PUSH:CCR NODY NODE@ DESTY! 10 TIMER!--ON A->DEST/TIME ;
8!
9! MAKE:MOVE DUP NOD VB! PUSH:CCR ROT MOVE:NODE ZEROTIMEB
10!2DUP NXTR VB! NXTC VB! 2DUP NODX NODE@ DESTX!
11!NODY NODE@ DESTY! SET:POSITION:XY A->DEST/TIME ;
12!
13! ON:TARGET? PUSH:CCR INTR VB@ = SWAP INTC VB@ = AND ;
14!-->
15!
```

FILE = CD

BLK= 3

0| (DELTA, DUMPER) -->

1| DD CR 8 0 DD

2| I . 2DUP MPLO I + NODEB@ . 2DUP NDXO I + NODEB@ .

3| 2DUP NDYO I + NODEB@ . CR LOOP 2DROP ;

4| -->

5|

6|

7|

8|

9|

10|

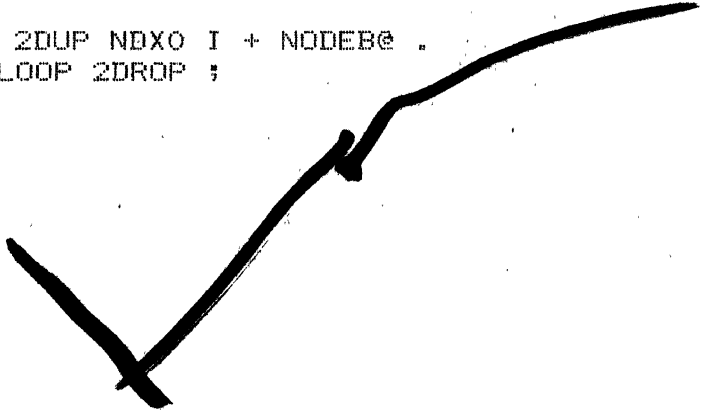
11|

12|

13|

14|

15|



FILE = CD

BLK= 0

```
01( COMPUTE DELTAS FOR STORAGE ROUTINE )
11( THIS ROUTINE COMPUTES DELTA FOR ONE COORDINATE )
21SUBR CDEL1 ( DE=R,C B=COORD PTR, C=DIR )
31B PUSH, D PUSH,
41B PUSH, C A MOV, MPLO ADI, A C MOV, node^ CALL, M L MOV,
51O H MVI, B POP, L A MOV, A ANA, O<>, IF,
61H PUSH, D PUSH, move:node CALL,
71B C MOV, node^ CALL, M E MOV, H INX, M D MOV, XCHG,
81XTHL, XCHG, node^ CALL, M C MOV, H INX, M B MOV,
91H POP, ( TARGET ) D POP, ( TIME ) CDELTA CALL, E A MOV,
101THEN, D POP, B POP, A B MOV, RET,
111-->
121
131
141
151
```

BLK= 1

```
01( SET DELTAS FOR BOTH COORDINATES FOR A GIVEN PATH )
11SUBR SETDELTS
21NBX B MVI, CDEL1 CALL, B PUSH, C A MOV, NDXO ADI, A C MOV,
31node^ CALL, B M MOV, B POP, NBY B MVI, CDEL1 CALL,
41B PUSH, C A MOV, NDYO ADI, A C MOV,
51node^ CALL, B M MOV, B POP, RET,
61-->
71
81
91
101
111
121
131
141
151
```

BLK= 2

```
01( COMPUTE DELTAS FOR WHOLE MATRIX )
11F= MAKELP
21CODE MAKEDELTS <ASSEMBLE B PUSH,
31O D LXI, O C MVI,
41LABEL MAKELP SETDELTS CALL,
51C A MOV, A INR, A C MOV, S CPI, MAKELP JRNZ, O C MVI,
61E A MOV, A INR, A E MOV, NCOLS CPI, MAKELP JRNZ, O E MVI,
71D A MOV, A INR, A D MOV, NROWS CPI, MAKELP JRNZ,
81B POP, NEXT ASSEMBLE>
91: FIXVGER NCOLS O DO NROWS O DO
101J I NODX NODE@ XADJ J I NBX NODE!
111J I NODY NODE@ YADJ J I NBY NODE! LOOP LOOP ;
121: MD FIXVGER MAKEDELTS ;
131-->
141
151
```


FILE = VR

BLK= 0

```
0:( HOPPED UP 8 BIT MPY ROUTINE )
1:( THIS ROUTINE IS USED TO MULTIPLY DELTA BY DISTANCE )
2:( ADDING RESULT TO INITIAL DISP )
3:( HL= INITIAL DISP, DE= DELTA, A= DIST )
4:SUBR HOTMPY RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
5:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
6:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
7:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
8:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
9:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
10:RRC, CY, IF, D DAD, THEN, E SLAR, D RALR,
11:RRC, CY, IF, D DAD, THEN, RET,
12:SUBR SQUARE BABS CALL, A E MOV, O D MVI, O H LXI,
13:HOTMPY JMPR,
14:-->
15:
```

BLK= 1

```
0:( CALCULATE X Y POSITION OF OBJECT FROM DISTANCE, BASE, AND )
1:( DELTAS )
2:SUBR CALCXY O C MVI,
3:INWR Y A LDX, MEMR Y CMPX, O<>, IF, C INR, A MEMR Y STX, THEN,
4:INWC Y A LDX, MEMC Y CMPX, O<>, IF, C INR, A MEMC Y STX, THEN,
5:INWD Y A LDX, MEMD Y CMPX, O<>, IF, C INR, A MEMD Y STX, THEN,
6:IDISTANCE 1+ Y A LDX, A B MOV, MEMDIST Y CMPX, O<>, IF,
7:IC INR, A MEMDIST Y STX, THEN,
8:IC A MOV, A ANA, O=, IF,
9:IVX Y E LDX, VX 1+ Y D LDX,
10:IVY Y L LDX, VY 1+ Y H LDX,
11:RET,
12:THEN, VBSUPDATE VLOGICSTAT Y SETX,
13:-->
14:
15:
```

BLK= 2

```
0:( MORE CUTE CALCULATIONS )
1:BA MOV,
2:BASEX Y L LDX, BASEX 1+ Y H LDX, DELTAX Y E LDX,
3:DELTAX 1+ Y D LDX, HOTMPY CALL, L VX Y STX, H VX 1+ Y STX,
4:HPUSH,
5:BASEY Y L LDX, BASEY 1+ Y H LDX, DELTAY Y E LDX,
6:DELTAY 1+ Y D LDX, HOTMPY CALL, L VY Y STX, H VY 1+ Y STX,
7:IDPOP,
8:RET,
9:-->
10:
11:
12:
13:
14:
15:
```

FILE = VR.

BLK= 3

```
0!( SET BASE POSITION )
1!( IN DE=ROW,COL )
2!SUBR SETBASEPOS B PUSH, D PUSH,
3!NBX C MVI, node^ CALL, M E MOV, H INX, M D MOV, H INX,
4!M C MOV, H INX, M B MOV,
5!E BASEX Y STX, D BASEX 1+ Y STX,
6!E VX Y STX, D VX 1+ Y STX,
7!C BASEY Y STX, B BASEY 1+ Y STX,
8!C VY Y STX, B VY 1+ Y STX,
9!D POP, B POP, RET,
10!SUBR FREEZEBASE A XRA,
11!A DISTANCE Y STX, A DISTANCE 1+ Y STX,
12!( A ACCDIST Y STX, A ACCDIST 1+ Y STX, )
13!A DELTADIST Y STX, A DELTADIST 1+ Y STX, RET,
14!-->
15!
```

BLK= 4

```
0!( ROUTINE TO ESTABLISH NEW BASE POSITIONS AND DELTAS )
1!( FIRST A SIGN ROUTINE )
2!SUBR SGNA A ANA, O A MVI, RP, A DCR, RET,
3!SUBR NEWPATH
4!NOWR Y D LDX, NOWC Y E LDX,
5!SETBASEPOS CALL, NOWD Y A LDX, MPLO ADI, A C MOV,
6!node^ CALL, M A MOV, A MAXDIST Y STX, S D LXI, D DAD,
7!M A MOV, A DELTAX Y STX, SGNA CALL, A DELTAX 1+ Y STX,
8!D DAD, M A MOV, A DELTAY Y STX, SGNA CALL, A DELTAY 1+ Y STX,
9!RET,
10!-->
11!
12!
13!
14!
15!
```

151K

BLK= 5

```
0!( ROUTINE TO CAUSE OBJECT TO ARRIVE AT A NEW POSITION )
1!SUBR ARRIVE DI,
2!NOWR Y D LDX, NOWC Y E LDX, NOWD Y C LDX,
3!move:node CALL, D NOWR Y STX, E NOWC Y STX,
4!SETBASEPOS CALL, FREEZEBASE CALL,
5!RET,
6!-->
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

*260
M260
160*

FILE = VR

BLK= 6

```
0!( DISTANCE PHASE ACCUMULATOR )
1!( DISTANCE HAS BOTH DELTA AND ACCELERATION )
2!( IN A= TIMEBASE TO USE )
3!SUBR DISTPA TBDEST TCHGSTAT Y BITX, RNZ,
4!DISTANCE Y L LDX, DISTANCE 1+ Y H LDX,
5!DELTADIST Y E LDX, DELTADIST 1+ Y D LDX,
6!( ACCDIST Y C LDX, ACCDIST 1+ Y B LDX, )
7!BEGIN, D DAD, ( XCHG, B DAD, XCHG, ) A DCR, O=, END,
8!( IF BEYOND MAX DISTANCE, SET AT MAX DISTANCE AND FLAG )
9!MAXDIST Y A LDX, A ANA, O<>, IF, H A MOV, MAXDIST Y CMPX,
10!CY~, IF, TBDEST TCHGSTAT Y SETX, MAXDIST Y H LDX, O L MVI,
11!THEN, THEN, E DELTADIST Y STX, D DELTADIST 1+ Y STX,
12!L DISTANCE Y STX, H DISTANCE 1+ Y STX,
13!RET,
14!-->
15!
```

BLK= 7

```
0!( DISTANCE VECTORING ROUTINE AND VGER VERBS )
1!5 C= TB-DVECT ( TVMROPT2 BIT TO ACTIVATE DISTANCE VECTORING )
2!SUBR DISTVECT PSW PUSH, B PUSH,
3!B A MOV, DISTPA CALL,
4!CALCX Y CALL, B POP, PSW POP, RET,
5!SUBR NEWVECT TB-DVECT TVMROPT2 Y BITX, vect JZ,
6!H PUSH, CUSVEC Y L LDX, CUSVEC 1+ Y H LDX, XTHL, RET,
7!HEX NEWVECT 89D9 ( 8956 ) U! DECIMAL ( ***** STUFF IN LINK )
8!CODE DVECT-OFF Y PUSHX, vaddr LIYD, TB-DVECT TVMROPT2 Y RESX,
9!Y POPX, NEXT
10!CODE DVECT-ON Y PUSHX, vaddr LIYD,
11!DISTVECT H LXI, L CUSVEC Y STX, H CUSVEC 1+ Y STX,
12!TB-DVECT TVMROPT2 Y SETX, Y POPX, NEXT
13!-->
14!
15!
```

BLK= 8

```
0!( CODE FOR TASKS TO INTERFACE TO NEW GOODIES )
1!CODE ESTPOS DI, B PUSH, Y PUSHX, vaddr LIYD,
2!NOWC Y E LDX, NOWR Y D LDX,
3!SETBASEPOS CALL,
4!FREEZEBASE CALL,
5!Y POPX, B POP, NEXT
6!( TRAVEL AWAY FROM NODE )
7!CODE DEPART:NODE DI, B PUSH, Y PUSHX, vaddr LIYD,
8!NEWPATH CALL,
9!Y POPX, B POP, NEXT
10!( ARRIVE NODE )
11!CODE ARRIVE:NODE DI, B PUSH, Y PUSHX, vaddr LIYD,
12!ARRIVE CALL,
13!Y POPX, B POP, NEXT
14!-->
15!
```

FILE = VR

BLK= 9

```
01( REVERSE DIRECTION ROUTINE )
1!SUBR REVERSE:DIRECTION
2!NOWR Y D LDX, NOWC Y E LDX, NOWD Y C LDX,
3!move:node CALL, C A MOV, CMA, 7 ANI,
4!D NOWR Y STX, E NOWC Y STX, A NOWD Y STX,
5!NEWPATH CALL, MAXDIST Y H LDX, O L MVI,
6!DISTANCE Y E LDX, DISTANCE 1+ Y D LDX,
7!A ANA, D DSBC, L DISTANCE Y STX, H DISTANCE 1+ Y STX,
8!RET,
9!CODE RUSH:SOURCE DI, B PUSH, Y PUSHX, vaddr LIYD,
10!DISTANCE Y A LDX, DISTANCE 1+ Y ORAX, O<>, IF,
11!REVERSE:DIRECTION CALL,
12!THEN,
13!Y POPX, B POP, NEXT
14!-->
15!
```

FILE = WR

BLK= 0

```
0:( VMR      SLEZR2A )
1:HEX
2:SUBR SLEZR2A ( does pat offset and relabs )
3: ( in- BC= magic/exp , HL= y , DE= x , IX= pattern addr )
4: ( out- HL= new vscradr , C= new vmagic )
5: invertxy? CALL, L SLAR, H RALR, L SLAR, H RALR, ( *4 for y )
6: invert? CALL,
7: H PUSH, XCHG, O X D LDX, O E MVI, ( x offset )
8: D SRAR, E RARR, D SRAR, E RARR, ( /4 for x offset )
9: MRFLOP C BIT, 0<>, IF, D DAD, ELSE, A ORA, D DSBC, THEN,
10: XTHL, ( push X+off, HL<-Y ) 1 X D LDX, O E MVI, ( y offset )
11: MRFLIP C BIT, 0<>, IF, D DAD, H DCX,
12: ELSE, A ORA, D DSBC, THEN,
13: D POP,
14:-->
15:
```

BLK= 1

```
0:( VMR )
1: ( y can not get here larger than 256 )
2: H A MOV, O H MVI, A L MOV, H DAD, H DAD, H DAD,
3: H DAD, D PUSH, L E MOV, H D MOV, H DAD, H DAD, ( *64 )
4: D DAD, ( *80 ) XCHG, H POP, ( x )
5: L A MOV, ( SAVE BIT CNT ) H L MOV, O H MVI, D DAD, ( x+y )
6: RLC, RLC, 3 ANI,
7: MRFLOP C BIT, 0<>, IF, NEG, 0=, IF, H DCX, THEN, THEN,
8: 3 ANI, A E MOV, invert? CALL, C A MOV, FC ANI, E ORA,
9:A C MOV, ( HL= screen address ) RET,
10:DECIMAL -->
11:
12:
13:
14:
15:
```

BLK= 2

```
0:( MY OWN EASY TO USE WRITE ROUTINE )
1:BV= INTERSTAT
2:CODE WRITEP A XRA, INTERSTAT STA, INTCPT IN,
3:IX PUSHX, D POP, EXX, X POPX, B POP, H POP, yadj CALL, XTHL,
4:yadj CALL, XCHG, H POP,
5:SLEZR2A CALL, X INXX, X INXX, O X E LDX, X INXX, O X D LDX,
6:IX INXX, write CALL, EXX,
7:INTCPT IN, INTERSTAT STA,
8:D PUSH, X POPX, NEXT
9:DECIMAL -->
10:
11:
12:
13:
14:
15:
```

FILE = SC

BLK= 0

```
01( SCORING GOODIES )
11
21RAMMARK SLENGTH R= P1SV RAMLEN C= P1SL VARHERE C= P1SS
31RAMMARK SLENGTH R= P2SV RAMLEN C= P2SL VARHERE C= P2SS
412 A= P1SCR 2 A= P2SCR
519 BA= AP1SCR 9 BA= AP2SCR
61: CLEAR:SCORE:VECTORS 0 P1SS P1SL FILL 0 P2SS P2SL FILL ;
71: CLEAR:SCORES 0 P1SCR ZERO 1 P1SCR ZERO
810 P2SCR ZERO 1 P2SCR ZERO CLEAR:SCORE:VECTORS ;
91-->
101
111
121
131
141
151
```

BLK= 1

```
01( TASK TO DISPLAY PLAYER ONES SCORE )
11: DISPP1SCR ;TASK:
210 P1SCR @ 1 P1SCR @ 1 AP1SCR 7 BIN->ASC
318 0 AP1SCR B! 48 1 AP1SCR B!
410 AP1SCR OSUPR
51-160 X! 99 Y!
61PLOP-ON
717 XPAND!
810 AP1SCR ANIM!
91STRING ;
101
111: BUMPP1SCR 0 P1SCR @ 1 P1SCR @ ROT 0 D+ 1 P1SCR ! 0 P1SCR !
121P1SV DISPP1SCR ;
131
141-->
151
```

BLK= 2

```
01( TASK TO DISPLAY PLAYER TWOS SCORE )
11: DISPP2SCR ;TASK:
210 P2SCR @ 1 P2SCR @ 1 AP2SCR 7 BIN->ASC
318 0 AP2SCR B! 48 1 AP2SCR B!
410 AP2SCR OSUPR
5198 X! 99 Y!
61PLOP-ON
717 XPAND!
810 AP2SCR ANIM!
91STRING ;
101: BUMPP2SCR 0 P2SCR @ 1 P2SCR @ ROT 0 D+ 1 P2SCR ! 0 P2SCR !
111P2SV DISPP2SCR ;
121: INCSCORE PLAYERUP @ IF BUMPP2SCR ELSE BUMPP1SCR THEN ;
131-->
141
151
```

FILE = SC

BLK= 3

```
0!( TOGGLE:LIFE, DISPLAY REMAINING LIVES, AND BITE DUST )
1!: TOGGLE:LIFE INITIAL-LIVES @ -2 / + 16 *
2!90 96 PAC8 WRITEP ;
3!
4!: DISPLAY:REMAINING:LIVES
5!REMAINING-LIVES @ 1- DUP IF
6!0 DO I TOGGLE:LIFE LOOP
7!ELSE DROP THEN ;
8!
9!: BITE:DUST REMAINING-LIVES 1-!
10!REMAINING-LIVES @ DUP IF 1- TOGGLE:LIFE
11!ELSE DROP 1 GAME-OVER ! STOPme 1+B! THEN ;
12!
13!-->
14!
15!
```

FILE = NGM

BLK= 0

```
0!( NEW CONFLICT CHECKER IN: DE=R,C B=D OUT: A= FLAG )
1!DATA CONCM 1 B, 0 B, 1 B, 0 B, 0 B, 6 B, 0 B, 6 B,
2!5 B, 0 B, 7 B, 0 B, 0 B, 0 B, 0 B, 2 B,
3!SUBR CONFLICT? B PUSH, 0 B MVI, CONCM H LXI, B DAD,
4!M A MOV, A ANA, 0=, IF, B POP, RET, THEN,
5!D PUSH, H PUSH, A C MOV, move:node CALL,
6!H POP, 8 B LXI, B DAD, M A MOV, MPLO ADI, A C MOV,
7!node^ CALL, M A MOV, D POP, B POP, RET,
8!CODE CONFLICT:CHECK EXX, B POP, H POP, D POP, L D MOV,
9!CONFLICT? CALL, A L MOV, 0 H MVI, H PUSH, EXX, NEXT
10!
11!( CHECK FOR LEGAL NODE )
12!( D= ROW, E= COL RETURNS CY SET IF LEGAL COMBO )
13!SUBR movecheck
14!D A MOV, NROWS CPI, RNC, E A MOV, NCOLS CPI, RET, -->
15!
```

BLK= 1

```
0!( VARIABLES FOR MATRIX GENERATOR )
1!V= GMRC V= GMD V= GMNRC
2!V= RCX V= RCY V= NRCX V= NRCY
3!-->
4!
5!
6!
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

BLK= 2

```
0!( ADD PATH ROUTINE )
1!SUBR addepath GMRC SDED, C A MOV, GMD STA, ( STUFF STUFF )
2!MPLO ADI, A C MOV, node^ CALL, M A MOV, A ANA, RNZ,
3!GMD LDA, A C MOV, move:node CALL, GMNRC SDED, ( SET NEW R, C )
4!movecheck CALL, RNC,
5!GMRC LDED, CONFLICT? CALL, A ANA, RNZ,
6!TOTAL-PATHS LHLD, H INX, TOTAL-PATHS SHLD, ( BUMP PATHS )
7!( COMPUTE DISTANCES AND DELTAS )
8!NODX C MVI, GMRC LDED, node^ CALL,
9!M E MOV, H INX, M D MOV, H INX, RCX SDED,
10!M E MOV, H INX, M D MOV, RCY SDED,
11!GMNRC LDED, node^ CALL,
12!M E MOV, H INX, M D MOV, H INX, NRCX SDED,
13!M E MOV, H INX, M D MOV, NRCY SDED,
14!-->
15!
```


FILE = NGM

BLK= 3

```
01( COMPUTE DISTANCE )
11RCY LHLD, A ANA, D DSBC, L A MOV, SQUARE CALL, H PUSH,
21NRCX LDED, RCX LHLD, A ANA, D DSBC, L A MOV,
31SQUARE CALL, D POP, D DAD, sqrt CALL, A B MOV, ( B= DIST )
41GMRC LDED, GMD LDA, MPLO ADI, A C MOV, node^ CALL, B M MOV,
51#CON C MVI, node^ CALL, M INR,
61GMD LDA, CMA, 7 ANI, MPLO ADI, A C MOV,
71GMNRC LDED, node^ CALL,
81B M MOV, #CON C MVI, node^ CALL, M INR, RET,
91CODE ADD:PATH EXX, B POP, H POP, D POP, L D MOV,
10|addrpath CALL, EXX, NEXT
11|-->
12|: ADD:PATH TOTAL-PATHS @ . 3 NDUP . . . CR ADD:PATH ;
13|-->
14|
15|
```

BLK= 4

```
01( ASSM CONNECTIVITY MARKER )
11BV= MAKCON
21F= MRPT F= MCLP F= MDLP F= NOSH F= NXTRC
31CODE MARK:CONNECTIVITY <ASSEMBLE EXX,
41LABEL MRPT A XRA, MAKCON STA, 0 D LXI,
51LABEL MCLP CONFLG C MVI, node^ CALL, M A MOV, A ANA,
61NXTRC JRNZ, ( SKIP IF ALREADY CONNECTED )
71MPLO CONFLG - B LXI, B DAD, 0 B MVI, ( B= DIR CTR )
81LABEL MDLP M A MOV, A ANA, NOSH JRZ, ( KICKOUT NOT REL )
91B C MOV, H PUSH, D PUSH,
10|move:node CALL, ( GOTO NEIGHBOR )
11|CONFLG C MVI, node^ CALL, D POP, M A MOV, H POP,
12|A ANA, ( IS NEIGHBOR MARKED? ) NOSH JRZ,
13|CONFLG C MVI, node^ CALL, 1 A MVI, A M MOV, MAKCON STA,
14|TOTAL-CONNECTS LHLD, H INX, TOTAL-CONNECTS SHLD,
15|NXTRC JMPR, -->
```

BLK= 5

```
01( TRY THE NEXT DIRECTION )
11LABEL NOSH B INR, H INX, B A MOV, 8 CPI, MDLP JRNZ,
21( GOTO NEXT GROTTTO )
31LABEL NXTRC E INR, E A MOV, NCOLS CPI, MCLP JRNZ, 0 E MVI,
41D INR, D A MOV, NROWS CPI, MCLP JRNZ,
51( KEEP SCANNING UNTIL THANGS STABILIZED )
61MAKCON LDA, A ANA, MRPT JRNZ, EXX, NEXT
71ASSEMBLE>
81-->
9|
10|
11|
12|
13|
14|
15|
```

FILE = GM

BLK= 0

```
0|( CONNECTIVITY TESTING )
1|: ZAM BKGV vaddr ! NCOLS 0 DO NROWS 0 DO J I
2|COMP:XY J I NODY NODE! J I NODX NODE! LOOP LOOP ;
3|: NOTE:CONNECTIVE CONFLG NODE^ BONE ;
4|: TEST:CONNECTIVE CONFLG NODEB@ ; -->
5|-->
6|
7|
8|
9|
10|
11|
12|
13|
14|
15|
```

BLK= 1

```
0|( CONNECT INDICATED ZONES TOGETHER )
1|: CRND DUP 0= IF 5 RND ELSE DUP NCOLS 1- = IF 5 RND 3 +
2|ELSE 8 RND THEN THEN ;
3|: ADD:ANOTHER TOTAL-PATHS @ BEGIN NCOLS 2 - RND 1+
4|NROWS 2 - RND 1+ CRND ADD:PATH DUP TOTAL-PATHS @
5|<> END DROP ;
6|: MAKE:MAZE CLEAR:NODEMAT ZAM
7|1 TOTAL-CONNECTS !
8|NCOLS 2 - RND 1+ DUP START-COL ! 0 NOTE:CONNECTIVE
9|NCOLS 2 - RND 1+ STOP-COL !
10|NCOLS 0 DO NROWS 0 DO J I CRND ADD:PATH LOOP LOOP
11|BEGIN
12|1 ( INIT ) NCOLS 0 DO NROWS 0 DO J I #CON NODEB@ 2 < IF
13|J I CRND ADD:PATH DROP 0 THEN LOOP LOOP END
14|BEGIN MARK:CONNECTIVITY TOTAL-CONNECTS @ 1 = WHILE
15|START-COL @ 0 CRND ADD:PATH REPEAT -->
```

BLK= 2

```
0|( KEEP COOKING UNTIL EVERYONES CONNECTED )
1|BEGIN
2|NCOLS 0 DO NROWS 0 DO J I TEST:CONNECTIVE NOT IF
3|J I CRND ADD:PATH THEN LOOP LOOP
4|MARK:CONNECTIVITY TOTAL-CONNECTS @ NNODES =
5|END
6|4 GAME# B@ 4 MIN - 4 * DUP IF 0 DO ADD:ANOTHER LOOP
7|ELSE DROP THEN ;
8|
9|( ARE WE IN THE START CHAMBER )
10|: START:CHAMBER?
11|2DUP START-ROW = IF START-COL @ = IF 2DROP 0 ELSE 1 THEN
12|ELSE DROP 1 THEN ;
13|-->
14|
15|
```

FILE = LD

BLK= 0

```
0:( **** LOCAL DISTANCE **** )
1:( LOCAL DISTANCE ROUTINE )
2:( THIS ROUTINE COMPUTES THE DISTANCE BETWEEN TWO OBJECTS )
3:( IN: IX= FOLLOWER IY= LEADER OUT: A=DIST, B= REV FLAG )
4:F= DIFB F= TRYM F= SAMD F= INFIN
5:SUBR LDIST <ASSEMBLE
6:NOWC X E LDX, NOWR X D LDX,
7:( DOES CI=CO AND RI=RO ? )
8:E A MOV, NOWC Y CMPX, TRYM JRNZ,
9:D A MOV, NOWR Y CMPX, TRYM JRNZ,
10:( ME AND HIM BOTH HAVE SAME ORIGIN )
11:( ARE WE ON THE SAME BRANCH? )
12:NOWD X A LDX, NOWD Y CMPX, DIFB JRNZ,
13:(. YES SIR - WE ARE ON SAME BRANCH )
14:DISTANCE 1+ Y A LDX, DISTANCE 1+ X SUBX, 0 B MVI, BABS JMP,
15!-->
```

BLK= 1

```
0:( WE ARE ON DIFERENT BRANCHES OF THE SAME ORIGIN )
1:LABEL DIFB DISTANCE 1+ Y A LDX,
2:DISTANCE 1+ X ADDX, 1 B MVI, BABS JMP,
3:LABEL TRYM NOWD X C LDX, H PUSH, move:node CALL, ( TO DEST )
4:H POP, MAXDIST X A LDX, DISTANCE 1+ X SUBX, ( REVERSE DIST )
5:A B MOV, ( AND SAVE IT IN B )
6:D A MOV, NOWR Y CMPX, INFIN JRNZ,
7:E A MOV, NOWC Y CMPX, INFIN JRNZ,
8:C A MOV, CMA, 7 ANI, NOWD Y CMPX, SAMD JRZ,
9:( I AM ON A PATH LEADING ME TO OTHERS ORIGIN )
10:B A MOV, DISTANCE 1+ Y ADDX, 0 B MVI, BABS JMP,
11:( I AM ON COMPLEMENTARY PATH THAT OBJECT IS ON )
12:LABEL SAMD DISTANCE 1+ Y A LDX, B SUB, 1 B MVI, BABS JMP,
13:( OBJECTS ARE FARTHER THEN WE CAN EASILY DETERMINE )
14:LABEL INFIN 127 A MVI, RET,
15:ASSEMBLE> -->
```

BLK= 2

```
0:( DISTANCE ROUTINE FOR LIST REFORMER TO USE )
1:( IF IT GETS INFINITY BACK IT WILL TRY SWAPPING X AND Y )
2:
3:SUBR LRDIST LDIST CALL, ( TRY IT ONE WAY )
4:127 CPI, RNZ, ( RETURN IF NON INFINITE )
5:( ITS INFINITE SO TRY IT THE OTHER WAY AROUND )
6:X PUSHX, XTIY, X POPX, LDIST CALL,
7:( BUT SWITCH BACK TO OLD POINTER SCAM BEFORE GOING HOME )
8:X PUSHX, XTIY, X POPX, RET,
9!-->
10:
11:
12:
13:
14:
15:
```

FILE = LD

BLK= 3

```
01( NEW FINDCLOSE ROUTINE )
11DECIMAL
21F= SRCL F= FCLD
31SUBR FINDCLOSE <ASSEMBLE
410 HOSTAB H LXI, EXX, 127 C MVI, EXX,
51LABEL SRCL M E MOV, H INX, M D MOV, H INX, D A MOV, E ORA,
61FCLD JRZ, D PUSH, X POPX, ASSMSV X A LDX, ASNOT CPI,
71SRCL JRNZ, HOSSV X A LDX, HSATP CPI, SRCL JRNZ,
81LDIST CALL, EXX, C CMP, CY, IF, A C MOV,
91X PUSHX, H POP, EXX, B A MOV, EXX, A B MOV, THEN,
101EXX, SRCL JMPR,
111LABEL FCLD EXX, RET,
121ASSEMBLE>
131-->
141
151
```

BLK= 4

```
01( CHECK FINDCLOSE, AND IF FOUND LIGHT UP FOLLOWER )
11SUBR LOOKFOLLOWER ( SEARCH LIST ) FINDCLOSE CALL,
21C A MOV, MAXASSM CPI, ( IS FOLLOWER CLOSE ENUF? )
31RNC, ( KICKOUT IF TOO FAR AWAY )
41DISPF Y CMPX, RC, ( OR TOO CLOSE )
51H PUSH, X POPX, ( IX= FOLLOWER )
61Y PUSHX, D POP, ( DE= LEADER )
71( LINK HER IN ) L BEHIND Y STX, H BEHIND 1+ Y STX,
81E AHEAD X STX, D AHEAD 1+ X STX, ASSIM ASSMSV X MVIX,
91DELTADIST Y A LDX, A DELTADIST X STX,
101DELTADIST 1+ Y A LDX, A DELTADIST 1+ X STX,
111B A MOV, A ANA, RZ, ( NEED WE REVERSE FOLLOWER? )
121D PUSH, H PUSH, Y POPX, REVERSE:DIRECTION CALL, Y POPX, RET,
131SUBR LOOKASS BEHIND Y A LDX, BEHIND 1+ Y ORAX, RNZ, B PUSH,
141D PUSH, H PUSH, X PUSHX, LOOKFOLLOWER CALL,
151X POPX, H POP, D POP, B POP, RET, -->
```

FILE = 0T

BLK= 0

```
01( CHECK FOR ONTOP )
11F= ONTL
21SUBR ONTOP? <ASSEMBLE
31O HOSTAB H LXI, O C MVI,
41LABEL ONTL M E MOV, H INX, M D MOV, H INX,
51D A MOV, E ORA, RZ,
61D PUSH, X POPX, HOSSV X A LDX, HSATP CPI, ONTL JRNZ,
71B PUSH, LRDIST CALL, B POP, ONTOPLMT CPI, CY, IF,
811 C MVI, THEN, A B MOV, 127 CPI, O<>, IF,
91DIST-1 X SUBX,
101O=, IF, 1 C MVI, ELSE, O<, IF, 1 C MVI, THEN, THEN, THEN,
111B DIST-1 X STX, ONTL JMPR;
121ASSEMBLE>
131-->
141
151
```

BLK= 1

```
01( PLAYERS INTERRUPT LEVEL ONTOP CHECKER )
11SUBR PILOTR
21ASSMSV Y A LDX, A ANA, O<>, IF,
31ONTOP? CALL, C A MOV, A ANA, RNZ,
41ASCool ASSMSV Y MVIX, ( CLEAR ONTOP STATE )
51THEN, LOOKASS CALL, ( CHECK MY ASS )
61RET,
71SUBR PILOTC X PUSHX, PILOTR CALL, X POPX, RET,
81-->
91
101
111
121
131
141
151
```

BLK= 2

```
01( PROPOGATE LEADERS DELTA DOWN THRU LIST )
11( IY= LEADERS VECTOR )
21F= CDLP SUBR COPYDELTS <ASSEMBLE
31BEHIND Y E LDX, BEHIND 1+ Y D LDX,
41LABEL CDLP
51D A MOV, E ORA, RZ, D PUSH, X POPX,
61L DELTADIST X STX, H DELTADIST 1+ X STX,
71BEHIND X E LDX, BEHIND 1+ X D LDX, CDLP JMPR,
81ASSEMBLE>
91-->
101
111
121
131
141
151
```

FILE = OT

BLK= 3

```
0!( MAKE ALL MY FRIENDS HALT RIGHT NOW )
1!F= EHN F= RELP
2!SUBR HALTNOW <ASSEMBLE
3!DI, B PUSH, D PUSH, H PUSH, X PUSHX, Y PUSHX,
4!O HOSTAB H LXI, PLYRV Y LXIX,
5!LABEL RELP M E MOV, H INX, M D MOV, H INX,
6!D A MOV, E ORA, EHN JRZ, D PUSH, X POPX,
7!HOSSV X A LDX, HSATP CPI, RELP JRNZ,
8!A XRA, A BEHIND X STX, A BEHIND 1+ X STX,
9!A AHEAD X STX, A AHEAD 1+ X STX,
10!A DELTADIST X STX, A DELTADIST 1+ X STX,
11!ASNOT ASSMSV X MVIX,
12!LRDIST CALL, A DIST-1 X STX, RELP JMPR,
13!LABEL EHN A XRA, A BEHIND Y STX, A BEHIND 1+ Y STX,
14!Y POPX, X POPX, H POP, D POP, B POP, ASONTOP A MVI,
15!ASSMSV PLYRV + STA, RET, ASSEMBLE> -->
```

BLK= 4

```
0!( INTERFACES TO THE TERSE WORLD )
1!
2!CODE PROPDeltas DJ, X PUSHX, Y PUSHX, B PUSH,
3!vaddr LIYD,
4!DELTADIST Y L LDX, DELTADIST 1+ Y H LDX,
5!COPYDELTS CALL,
6!B POP, Y POPX, X POPX, NEXT
7!-->
8!
9!
10!
11!
12!
13!
14!
15!
```

FILE = HF

BLK= 0

```
0|( INTERFACES TO THE TERSE WORLD )
1|CODE JOIN:LINE DI, X PUSHX, Y PUSHX, B PUSH,
2|VADDR LIYD, HSATP HOSSV Y MVIX, FLYRV Y LXIX,
3|HALTNOW CALL,
4|B POP, Y POPX, X POPX, NEXT
5|
6|-->
7|
8|
9|
10|
11|
12|
13|
14|
15|
```

BLK= 1

```
0|( ASSIMULATED NODE ROUTINE )
1|F= GOHM,F= VIRG
2|SUBR HASSIM <ASSEMBLE DI, PSW PUSH,
3|DISTVECT CALL,
4|LOOKASS CALL,
5|VIRGIN Y A LDX, A ANA, O<, IF, O VIRGIN Y MVIX, VIRG JMPR,
6|THEN,
7|( AM I AT THE END OF THIS PATH? )
8|TBDEST TCHGSTAT Y BITX, GOHM JRZ, ( NO - KICKOUT )
9|-->
10|
11|
12|
13|
14|
15|
```

BLK= 2

```
0|( MORE )
1|LABEL VIRG
2|X PUSHX, H PUSH, D PUSH, B PUSH, ( GRAB PARMS FROM LDR )
3|NOWR B LXI, Y PUSHX, H POP, B DAD, XCHG,
4|AHEAD Y L LDX, AHEAD 1+ Y H LDX, ( HL= FL )
5|H PUSH, X POPX,
6|B DAD, POSLEN B LXI, LDIR,
7|( SET HOS DISTANCE TO N UNITS LESS THAN LEADER )
8|DISTANCE 1+ X A LDX, DISPF X SUBX, O<, IF, A XRA, THEN,
9|A DISTANCE 1+ Y STX, A XRA, A DISTANCE Y STX,
10|TBDEST TCHGSTAT Y RESX, ( DON'T ALARM TERSE )
11|B POP, D POP, H POP, X POPX,
12|LABEL GOHM PSW POP, RET, ASSEMBLE> -->
13|
14|
15|
```

FILE = HF

BLK= 3

```
01( FOLLOW MONSTER ROUTINE )
11SUBR MONFOLLOW DI, B PUSH,
21Y PUSHX, H POP, NOWR B LXI, B DAD, XCHG,
31SNATCHER Y L LDX, SNATCHER 1+ Y H LDX, B DAD,
41SNATLEN B LXI, LDIR, A XRA, A DELTADIST Y STX,
51A DELTADIST 1+ Y STX,
61CALCXY CALL,
71B POP, PSW POP, RET,
81-->
91
101
111
121
131
141
151
```

BLK= 4

```
01( SPECIAL MASTER VECTORING ROUTINE FOR HOSTAGES )
11
21SUBR HOSTAGEVECTOR PSW PUSH,
31HOSSV Y A LDX, HSATM CPI, MONFOLLOW JRZ,
41ASSMSV Y A LDX, A ANA,
51O<>, IF, PSW POP, HASSIM JMP,
61THEN, PSW POP, DISTVECT JMP,
71
81CODE HVECT-ON Y PUSHX, vaddr LIYD,
91HOSTAGEVECTOR H LXI, L CUSVEC Y STX, H CUSVEC 1+ Y STX,
101TB-DVECT TVMROPT2 Y SETX, Y POPX, NEXT
111-->
121
131
141
151
```


FILE = LFN

BLK= 0

```
0:( LOOK FOR NEARBY THANGS )
1:( HL= R,C IX= SUBJ RET Z IF NEAR, NZ IF NOT )
2:SUBR NEARBY? NOWR X D LDX, NOWC X E LDX,
3:D A MOV, H CMP, O=, IF, E A MOV, L CMP,
4:RZ, THEN,
5:DISTANCE 1+ X A LDX, A ANA, O=, IF, A INR, RET, THEN,
6:NOWD X C LDX, H PUSH, move:node CALL, H POP,
7:D A MOV, H CMP, RNZ, E A MOV, L CMP, RET,
8:
9:( NEARBY LIST -- HL/= TARG HL= LIST RET Z= NONE NZ= FOUND )
10:SUBR NEARBYLIST M E MOV, H INX, M D MOV, H INX,
11:D A MOV, E ORA, RZ, D PUSH, X POPX, EXX,
12:NEARBY? CALL, EXX, NEARBYLIST JRNZ,
13:I A MVI, A ANA, RET,
14!-->
15:
```

BLK= 1

```
0:( CODE ROUTINE TO DO NEARBY CHECK )
1:( C R LIST MTC? --- T )
2:CODE MTC? H POP, ( HL= LIST )
3:EXX, D POP, H POP, E H MOV, EXX, ( R,C )
4:X PUSHX, NEARBYLIST CALL, O H LXI, O=, IF, H INX, THEN,
5:X POPX, H PUSH, NEXT
6:
7:DATA PCONFT MONV1 , MONV2 , MONV3 , MONV4 , HOSV1 , HOSV2 ,
8:HOSV3 , HOSV4 , TRSV1 , TRSV2 , TRSV3 , TRSV4 , TV1 , 0 ,
9:
10: NOBODY:HOME:YET? 2DUP PCONFT MTC? IF 1 ELSE 2DROP 0 THEN ;
11!-->
12:
13:
14:
15:
```

FILE = T

BLK= 0

```
0!( PLACE TREASURE IN MAZE )
1!TABLE *TREASURE-MAP TRSV1 ; TRSV2 ; TRSV3 ; TRSV4 ; 0 ;
2!-->
3!
4!
5!
6!
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

BLK= 1

```
0!( TASK FOR A HUNK OF TREASURE )
1!
2! TREASURE-TASK ;TASK: 20 RND TIMER!--ON WAIT
3!( MAKE SELF APPEAR )
4!ESTPOS
5!THEJEWEL ANIM! 1STWRITE
6!XOR-ON ZERODXDYAXAY
7!10 TIMEBSALE!
8!SELF MYFLAG V^ FLAG!--ON GO DI ( TREA-S ) ZEROTIMEB
9!2000 INCSCORE NULPAT ANIM! 1 TIMER!--ON GO ;
10!-->
11!
12!
13!
14!
15!
```

BLK= 2

```
0!( PLACE TREASURE IN MAZE )
1!V= THESPOT
2! HIDE:PEICE THESPOT ! BEGIN BEGIN
3!NCOLS RND NROWS RND START:CHAMBER? END
4!NOBODY:HOME:YET? END
5!2DUP THESPOT @ NOWR OVB! THESPOT @ NOWC OVB!
6!THESPOT @ ROLL >TREASURE NODE! THESPOT @ TREASURE-TASK ;
7! HIDE:TREASURE TOTAL-JEWELS 0 DO
8!I TREASURE-MAP @ HIDE:PEICE LOOP ;
9! TREASURE:CHECK PUSH:CCR >TREASURE NODE@ DUP IF
10!DUP MYTYPE OVB@ T-TYP = IF
11!( JEWELS-REVEALED 1+! ) THEN 1 SWAP MYFLAG OVB!
12!O PUSH:CCR >TREASURE NODE! ELSE DROP THEN ;
13!S
14!
15!
```

FILE = RS

BLK= 0

```
0:( ROUTE SEARCH ROUTINE )
1:( VISITED MATRIX GOODIES )
2:DATA BITMASKS 1 B, 2 B, 4 B, 8 B, 16 B, 32 B, 64 B, 128 B,
3:SUBR BIT^ D PUSH, H PUSH, A E MOV, O B MVI, BITMASKS H LXI,
4:D DAD, M A MOV, H POP, D POP, RET,
5:SUBR VIS? H PUSH, B PUSH, Y PUSHX, H POP, VISMAT B LXI, B DAD,
6:E C MOV, B DAD, D A MOV, BIT^ CALL, M ANA, B POP, H POP, RET,
7:SUBR SETVIS H PUSH, B PUSH, Y PUSHX, H POP, VISMAT B LXI,
8:B DAD, E C MOV, B DAD, D A MOV, BIT^ CALL, M ORA, A M MOV,
9:B POP, H POP, RET,
10:( CLEAR OUT VIS BITMATRIX )
11:SUBR ZAPVIS B PUSH, H PUSH, VISMAT B LXI, Y PUSHX, H POP,
12:B DAD, NCOLS DO, O M MVI, H INX, LOOP, H POP, B POP, RET,
13:SUBR noded^ node^ CALL, D PUSH, MPLO D LXI, D DAD, D POP, RET,
14:-->
15:
```

BLK= 1

```
0:( GENERATE TREE ENTRYS FOR ONE ENTRY )
1:IF= RUGLP
2:SUBR GENTE <ASSEMBLE MPLO C MVI, node^ CALL, H PUSH, 8 B MVI,
3:LDAR, 7 ANI, A C MOV,
4:BEGIN, H POP, H PUSH, B A MOV, O B MVI, B DAD, A B MOV,
5:M A MOV, A ANA, O<>, IF, D PUSH, move:node CALL,
6:VIS? CALL, O=, IF, ( GENERATE NODE )
7:SETVIS CALL,
8:MYBOSS Y A LDX, A TPL X STX, MYBOSS 1+ Y A LDX, A TPL 1+ X STX,
9:E TC X STX, D TR X STX, C TD X STX,
10:TREECK Y L LDX, TREECK 1+ Y H LDX, FORKETH CALL, ( END CHECK? )
11:TEL D LXI, D DADX,
12:THEN, D POP, THEN, C A MOV, A INR, 7 ANI, A C MOV, LOOP, H POP,
13:RET,
14:ASSEMBLE>
15:-->
```

BLK= 2

```
0:( ADVANCE TREE ONE DEPTH DOWN )
1:SUBR ADVT MYBOSS Y L LDX, MYBOSS 1+ Y H LDX,
2:H INX, H INX, M E MOV, H INX, M D MOV,
3:GENTE CALL, MYBOSS Y L LDX, MYBOSS 1+ Y H LDX,
4:TEL D LXI, D DAD, M E MOV, H INX, M D MOV,
5:D INX, D A MOV, E ORA, O=, IF, H INX, ELSE, H DCX, THEN,
6:IL MYBOSS Y STX, H MYBOSS 1+ Y STX, ADVT JRNZ,
7:-1 X O MVIX, X INXX, -1 X O MVIX, X INXX, RET,
8:-->
9:
10:
11:
12:
13:
14:
15:
```

FILE = RS

BLK= 3

```
01( FIND PATH ROUTINE )
11( BC=TARGET R,C DE= NOWR,NOWC HL= ENDCHK IY= TREE RAM )
21CODE STARTSEARCH X PUSHX, D POP, Y PUSHX, H POP, EXX,
31H POP, vaddr LIYD, ZAPVIS CALL,
41A XRA,
51A FNDPTR Y STX, A FNDPTR 1+ Y STX,
61A MYBOSS Y STX, A MYBOSS 1+ Y STX,
71NOWR Y D LDX, NOWC Y E LDX,
81L TREECK Y STX, H TREECK 1+ Y STX,
91Y PUSHX, X POPX, TREES B LXI, B DADX,
101X PUSHX, GENTE CALL, H POP,
111L MYBOSS Y STX, H MYBOSS 1+ Y STX,
121-1 X O MVIX, X INXX, -1 X O MVIX, X INXX,
131X PUSHX, D POP, E FRONTIER Y STX, D FRONTIER 1+ Y STX,
141EXX, D PUSH, X POPX, H PUSH, Y POPX, NEXT -->
15!
```

BLK= 4

```
01( MORE PATH FINDER )
11F= TREELP F= SCANBK F= SCAN1
21SUBR BANGTREE <ASSEMBLE
31FRONTIER Y E LDX, FRONTIER 1+ Y D LDX, D PUSH, X POPX,
41FNDPTR Y L LDX, FNDPTR 1+ Y H LDX,
51L A MOV, H ORA, SCAN1 JRNZ, ADVT CALL,
61X PUSHX, D POP, E FRONTIER Y STX, D FRONTIER 1+ Y STX,
71A XRA, RET,
81-->
9!
10!
11!
12!
13!
14!
15!
```

BLK= 5

```
01( MORE )
11LABEL SCAN1 O B LXI,
21LABEL SCANBK M E MOV, C M MOV, H INX,
31M D MOV, B M MOV, H DCX, H B MOV, L C MOV,
41E A MOV, D ORA,
51O<>, IF, XCHG, SCANBK JMPR, THEN, 1 A MVI, A ANA, RET,
61ASSEMBLE>
7!
81CODE LOOKAHEAD Y PUSHX, D POP, X PUSHX, H POP, EXX,
91vaddr LIYD, BANGTREE CALL, O=, IF,
101O H LXI, ELSE, H PUSH, 1 H LXI, THEN, H PUSH,
111EXX, H PUSH, X POPX, D PUSH, Y POPX, NEXT
12!
131-->
14!
15!
```

FILE = RS

BLK = 6

```
0:( ROUTINE TO FIND BEST PATH TOWARDS TARGET )
1:( CHECK ROUTINE - ARE WE HOME YET? )
2:SUBR BULLSEYE? INTR Y A LDX, D CMP, RNZ,
3:INTC Y A LDX, E CMP, RNZ, X PUSHX, H POP,
4:L FNDPTR Y STX, H FNDPTR 1+ Y STX, RET,
5: RECON
6:BULLSEYE? STARTSEARCH BEGIN SYNC DI
7:LOOKAHEAD END TRACKPTR V! COGO ;
8:CODE FOLLOWTRACK Y PUSHX, vaddr LIYD,
9:TRACKPTR Y L LDX, TRACKPTR 1+ Y H LDX,
10:M E MOV, H INX, M D MOV, H INX, H INX, H INX,
11:E TRACKPTR Y STX, D TRACKPTR 1+ Y STX, M L MOV, O H MVI,
12:Y POPX, H PUSH, NEXT ASSEMBLE> -->
13:
14:
15:
```

FILE = H

BLK= 0

```
0|( HOSTAGE TABLE, HOSTAGE INTERCEPT CHECKER )
1|( CHECK HOSTAGE INTERCEPT WITH MONSTERS )
2|DATA MONLIST MONV1 , MONV2 , MONV3 , MONV4 , 0 ,
3|HEX 0202 DECIMAL C= XYHOST
4|( HOSTAGES INTERCEPT CHECKER, RUNS AS HOOK )
5|SUBR HOS-MON? FREEZE? CALL, RNZ, EXX,
6|MONLIST H LXI, XYHOST B LXI, CHECK:VECTOR:LIST CALL,
7|O>, IF,
8|1 MYFLAG Y MVIX, ( SET ME EATEN ) FREEZE CALL,
9|X PUSHX, D POP, E SNATCHER Y STX, D SNATCHER 1+ Y STX,
10|Y PUSHX, D POP, E MYSLAVE X STX, D MYSLAVE 1+ X STX,
11|HSATM HOSSV Y MVIX, HALTNOW CALL,
12|1 MYFLAG X MVIX, ( TELL MONSTER MOVE FLAG ) THEN,
13|EXX, RET,
14|-->
15|
```

BLK= 1

```
0|( TASK FOR A TEST HOSTAGE ) HEX 400 C= EXITVEL DECIMAL
1|( V= RECURADDR )
2|: HOSTAGE-TASK ;TASK: DI H-H-D DISPF VB! H-TYP MYTYPE VB!
3|ZEROTIMEB 20 RND TIMER!--ON WAIT DI 1STWRITE
4|ESTPOS ESTVALDIR BEGIN DI 0 MYFLAG VB!
5|HOSSV VB@ HSFREE CASE DVECT-ON
6|HOS10F ANIM! XOR-ON 10 TIMEBSCALE! 0 TIMEBMAX!
7|MYFLAG V^ FLAG!--ON GO
8|ELSE HSATP CASE
9|PRTBM TIMEBMAX!
10|HOS10 ANIM! JOIN:LINE
11|1 VIRGIN VB! 1 TIMEBSCALE!
12|MYFLAG V^ FLAG!--ON HOS-MON? HOOK!--ON
13|500 INCSCORE HVECT-ON GO
14|-->
15|
```

BLK= 2

```
0|( FOLLOW MONSTER TO NEW HANGOUT )
1|ELSE HSATM CASE FREEZETH
2|FLAG-OFF HVECT-ON
3|HOOK-OFF
4|ZEROTIMEB
5|( FOLLOW MONSTER TO ITS TARGET POSITION )
6|BEGIN MYFLAG V^ FLAG!--ON GO DI FLAG? END
7|ESTPOS ESTVALDIR
8|UNFREEZE HSFREE HOSSV VB! ASNOT ASSMSV VB!
9|ELSE DROP THEN THEN THEN 0 END ;
10|-->
11|
12|
13|
14|
15|
```

FILE = H

BLK= 3

```
0:( PLACE HOSTAGES IN MAZE )
1: HIDE:HOS THESPOT ! BEGIN BEGIN
2:NCOLS RND NROWS RND START:CHAMBER? END
3:NOBODY:HOME:YET? END
4:THESPOT @ NOWR OVB! THESPOT @ NOWC OVB!
5:THESPOT @ HOSTAGE-TASK ;
6: JAIL:HOS TOTAL-HOSTAGES 0 DO
7:I HOSTAB @ HIDE:HOS LOOP ;
8:S
9
10
11
12
13
14
15
```

FILE = R

BLK= 0

```
01( VGS interrupt vector erase  VERASE VERASEWRITE ) <STK
11SUBR XOR-FLIP VOXPAND Y B LDX, VOMAGIC Y C LDX,
21VOPATH Y H LDX,
31 VOPAT Y L LDX, H INX, H INX, ( pat off set) H PUSH, X POPX,
41 VOSCRADRH Y H LDX, VOSCRADR Y L LDX,
51 writer JMP, ( erase it )
61
71
81-->
91
101
111
121
131
141
151
```

BLK= 1

```
01( ROUTINE TO LINK TO VGER WRITE ROUTINE )
11SUBR WRITE-LINK
21 VBNOWRITE VLOGICSTAT Y BITX, 0=, IF, INTCPY IN, VWRITE CALL,
31 TBINTCPT-CHK TVMROPT Y BITX, 0<>, IF, INTCPY IN,
41 A ANA, 0<>, IF, TBINTCPT TCHGSTAT Y SETX,
51 TBNOVECT TVMROPT Y SETX, THEN, THEN,
61 ELSE, VBNOWRITE VLOGICSTAT Y RESX, THEN, RET, STK<> -->
71
81
91
101
111
121
131
141
151
```

BLK= 2

```
01( CHECK:NEAR )
11DATA PCON PLYRV , MONV1 , MONV2 , MONV3 ,
21MONV4 , TV1 , TRSV1 , TRSV2 , TRSV3 , TRSV4 ,
31HOSV1 , HOSV2 , HOSV3 , HOSV4 , 0 ,
41-->
51
61
71
81
91
101
111
121
131
141
151
```


FILE = R

BLK= 3

```
0!( SPECIAL WRITE ROUTINE FOR REVEALS )
1!HEX 0C0C C= XYZONE DECIMAL
2!F= REML F= RESL F= LISTEND
3!SUBR REVEALWRITE <ASSEMBLE 0 H LXI, H PUSH, ( MARK STACK )
4!( Y PUSHX, H POP, CONFTAB D LXI, D DAD, )
5!PCON H LXI,
6!LABEL REML M E MOV, H INX, M D MOV, H INX, D A MOV, E ORA,
7!LISTEND JRZ, D PUSH, X POPX,
8! VBNOERASE VLOGICSTAT X BITX, REML JRNZ,
9! VOPATH X A LDX, VOPAT X ORAX, REML JRZ,
10!
11!-->
12!
13!
14!
15!
```

BLK= 4

```
0!( MORE OF SPECIAL WRITE ROUTINE FOR REVEALS )
1!XYZONE B LXI,
2!PROXIMITY-CHECK CALL, REML JRZ,
3!X PUSHX, H PUSH, Y PUSHX, X PUSHX, Y POPX, XOR-FLIP CALL,
4!Y POPX, H POP, REML JMPR,
5!LABEL LISTEND WRITE-LINK CALL,
6!LABEL RESL D POP, D A MOV, E ORA, transition JZ,
7!Y PUSHX, D PUSH, Y POPX,
8!XOR-FLIP CALL, Y POPX, RESL JMPR,
9!ASSEMBLE> -->
10!-->
11!
12!
13!
14!
15!
```

BLK= 5

```
0!( TASK TO REVEAL PARTIAL PASSAGEWAY )
1!HEX
2!400 C= INITIAL#LEAP 8 C= MAXREVDIST
3!100 C= REVVEL 4 C= SHORTGOAL
4! : OTHER:REVEAL ;TASK: DI
5!ESTPOS DEPART:NODE INITIAL#LEAP DISTANCE V!
6!REVVEL DELTADIST V!
7!8 XPAND!--ON
8!REVEALPAT ANIM!
9!OR-ON PRIBM TIMEBMAX!
10!1STWRITE DVECT-ON
11!MAXREVDIST MAXDIST VB! REVEALWRITE ZGO ;
12!DECIMAL -->
13!
14!
15!
```

FILE = R

BLK= 6

```
0!( STUFF TO SENDOFF OTHER REVEALS )
1!( VECTOR# INDEX:VECTOR --- VECTORADDRESS )
2! INDEX:VECTOR REVL * RVOV SWAP - ;
3! START:OTHER:REVEALS
4! 0 DO PUSH:CCR I TEST:REL
5! IF PUSH:CCR I TEST:DRAWN
6! NOT IF
7! PUSH:CCR I INDEX:VECTOR NOWR OVB!
8! I INDEX:VECTOR NOWC OVB!
9! I I INDEX:VECTOR NOWD OVB!
10! I INDEX:VECTOR OTHER:REVEAL DI
11! THEN THEN LOOP ; -->
12!
13!
14!
15!
```

BLK= 7

```
0!( HEADLIGHT REVEALER )
1! HEX : HEADLIGHT:REVEAL ; TASK: DI
2! NOWC PLYRV OVB@ NOWC VB! NOWR PLYRV OVB@ NOWR VB!
3! NOWD PLYRV OVB@ NOWD VB! ESTPOS DEPART:NODE
4! MAXDIST VB@ SHORTGOAL - MAXDIST VB!
5! REVEALPAT ANIM! OC XPAND!--ON OR-ON 1STWRITE PRIBM TIMEBMAX!
6! INITIAL#LEAP DISTANCE V! REVVEL DELTADIST V! DVECT-ON
7! REVEALWRITE ZGO DI
8!-->
9!
10!
11!
12!
13!
14!
15!
```

BLK= 8

```
0!( MORE HEADLIGHT REVEALER )
1! PUSH:CCR NOWD VB@ TEST:DRAWN NOT IF
2! REVEALED-PATHS 1+! ( INCREMENT # OF PATHS REVEALED )
3! THEN
4! PUSH:CCR NOWD VB@ SET:DRAWN
5! ARRIVE:NODE PUSH:CCR NOWD VB@ COM 7 AND SET:DRAWN
6! PUSH:CCR TEST:GROTTO:DRAWN NOT IF
7! START:OTHER:REVEALS
8! GROTTO PAT ANIM! 1STWRITE OC XPAND!--ON
9!( CHECK:NEAR ) TOTAL-REVEALED-GROTTO 1+!
10! 1 TIMER!--ON REVEALWRITE ZGO PUSH:CCR SET:GROTTO:DRAWN THEN ;
11! DECIMAL -->
12!
13!
14!
15!
```

FILE = R

BLK= 9

```
0:( REVEAL FIRST CHAMBER )
1:HEX BV= UNROLL
2: INITIAL:REVEAL ;TASK:
3:PLYRV NOWR OVB@ NOWR VB!
4:PLYRV NOWC OVB@ NOWC VB! ESTPOS DVECT-ON
5:START:OTHER:REVEALS
6:GROTTOPAT ANIM! 1STWRITE OC XPAND! XPAND-ON OR-ON
7:1 TIMER!-ON REVEALWRITE ZGO PUSH:CCR SET:GROTTO:DRAWN
8:18 UNROLL B!
9:BEGIN 1 TIMER!-ON WAIT UNROLL B@ DUP VERBL OUTP 4 + DUP
10:UNROLL B! ODO = END ;
11:
12:DECIMAL -->
13:
14:
15:
```

FILE = K

BLK= 0

```
01( KEY MONITOR - WAIT FOR N CHAMBERS TO BE REVEALED )
11
21: KEY-TASK ;TASK: K-TYP MYTYPE VB!
31BEGIN 30 TIMER!-ON WAIT DI
41TOTAL-REVEALED-GROTTO @ KEY-THRESHOLD @ > END
51BEGIN BEGIN
61NCOLS RND NROWS 2- RND START:CHAMBER? END
71NOBODY:HOME:YET? END
81NOWR VB! NOWC VB!
91SELF PUSH:CCR >TREASURE NODE!
101-->
111
121
131
141
151
```

BLK= 1

```
01( KEY REVEALER )
11ESTPOS
21KEYPAT ANIM! 1STWRITE XOR-ON
31MYFLAG V^ FLAG!-ON DVECT-ON GO DI
41( KEY-S )
51( NOW REVEAL EXIT CHAMBER )
61BEGIN
71STOP-COL B@ NOWC VB! START-ROW NOWR VB! ESTPOS
81GROTTOPAT ANIM! PLEASE-UPDATE
91XOR-ON XPAND-ON 8 XPAND! 30 TIMER!-ON GO DI
101-->
111
121
131
141
151
```

BLK= 2

```
01( REVEAL THE EXIT CHAMBER )
11GROTTOPAT ANIM! 1STWRITE 12 XPAND! XPAND-ON OR-ON
21ESTPOS
312 TIMER!-ON REVEALWRITE ZGO DI
41BEGIN ESTPOS
51GROTTOPAT ANIM!
611STWRITE XOR-ON XPAND-ON 8 XPAND! 20 TIMER!-ON GO DI
71-->
81
91
101
111
121
131
141
151
```

FILE = K

BLK= 3

```
0:( MORE EXIT REVEALER AND KEY HIDER )
1:PLYRV NOWC OV@ NOWC VB@ = IF
2:PLYRV NOWR OV@ NOWR VB@ = IF
3:STOPme 1+B! NOBREAK BZERO
4:THEN THEN
5:0 END ;
6: HIDE:KEY BEGIN BEGIN
7:INCOLS RND NROWS 2- RND START:CHAMBER? END
8:NOBODY:HOME:YET? END
9:2DUP TV1 NOWR OV@ TV1 NOWC OV@
10:TV1 ROLL >TREASURE NODE! TV1 KEY-TASK ;
11:-->
12:
13:
14:
15:
```

BLK= 4

```
0:( ROUTINE TO END GAME )
1: END-GAME ;TASK:
2:0 BEHIND PLYRV OV@ BEGIN DUP WHILE SWAP 5000 + SWAP
3:BEHIND OV@ REPEAT DROP INCSCORE 60 TIMER!--ON WAIT
4:STOPme 1+B! NOBREAK BZERO ;
5:-->
6:
7:
8:
9:
10:
11:
12:
13:
14:
15:
```

FILE = P

BLK= 0

```
01( JOYSTICK ROUTINES )
11:HEX ( BV= JOYCODE BV= JOYLAST ) ( DSOO DP ! ***** )
21:DATA JOYTBL -1 B, -1 B, -1 B, -1 B, -1 B, 0 B, 5 B, -1 B,
3:-1 B, 2 B, 7 B, -1 B, -1 B, -1 B, -1 B, -1 B,
4:-1 B, 1 B, 6 B, -1 B, 3 B, 0 B, 5 B, -1 B,
5:4 B, 2 B, 7 B, -1 B, -1 B, -1 B, -1 B, -1 B,
6:( SUBR MYINTR PSW PUSH, H PUSH, 12 IN, CMA, 1F ANI,
7:JOYLAST H LXI, M CMF, A M MOV, 0<>, IF, 1F A MVI, THEN,
8:JOYCODE STA, H POP, PSW POP, SUI1 JMP, )
9:SUBR set:Joycode 12 IN, CMA, 1F ANI, A E MOV, 0 D MVI,
10:JOYTBL H LXI, D DAD, M A MOV, A ANA, RET,
11:CODE GET:JOYCODE
12:12 IN, CMA, 1F ANI, A E MOV, 0 D MVI, JOYTBL H LXI,
13:D DAD, M A MOV, A ANA, 0<>, IF, 0 H LXI, ELSE,
14:A E MOV, D PUSH, 1 H LXI, THEN, H PUSH, NEXT
15:DECIMAL -->
```

BLK= 1

```
01( NEW SCAN ADJUSTER )
11:DATA CCWTBL 3 B, 0 B, 1 B, 5 B, 2 B, 6 B, 7 B, 4 B,
21:DATA CWTBL 1 B, 2 B, 4 B, 0 B, 7 B, 3 B, 5 B, 6 B,
31F= scann F= nose
41SUBR adj-scan <ASSEMBLE
51H PUSH, 0 B MVI, B DAD, M A MOV, A ANA,
61scann JRZ, H POP, C A MOV, RET,
71LABEL scann CCWTBL H LXI, B DAD, M E MOV, 0 D MVI,
81H POP, H PUSH, D DAD, M D MOV,
91CWTBL H LXI, B DAD, C A MOV, M C MOV, H POP, B DAD,
10:A B MOV, M A MOV,
11:A ANA, 0<>, IF, D A MOV, A ANA, nose JRNZ,
12:C A MOV, RET, THEN, D ORA, nose JRZ, E A MOV, RET,
13:LABEL nose B A MOV, RET,
14:ASSEMBLE>
15|-->
```

BLK= 2

```
01( MORESTUFF )
11:CODE ADJ-SCAN EXX, B POP, H POP,
21adj-scan CALL, A L MOV, 0 H MVI, H PUSH, EXX, NEXT -->
31: TST B 0 DO 2DUP MPLO NODE^ I DUP . ADJ-SCAN . CR LOOP 2DROP :
41-->
51
61
71
81
91
101
111
121
131
141
151
```

SVRP

FILE = P

BLK= 3

```
0:( INTERRUPT LEVEL JOY MONITOR )
1:BV= OBJECT-MOVING
2:SUBR JOYCHECK OBJECT-MOVING LDA, A ANA, RZ,
3:ITBDEST TCHGSTAT Y BITX, RNZ, DISTANCE 1+ Y A LDX, A ANA, RZ,
4:set:joycode CALL,
5:O<, IF, ASSMSV Y A LDX, ASCOOL CPI, O<>, IF,
6:O H LXI, PLYRV DELTADIST + SHLD, THEN, RET,
7:THEN, PLAYERVELO LHLD, PLYRV DELTADIST + SHLD, CMA, 7 ANI,
8:NOWD Y CMPX, RNZ,
9:REVERSE:DIRECTION CALL, HALTNOW CALL,
10:NOWD Y A LDX, RRC, RRC, RRC, A VANGLE Y STX, RET,
11:SUBR PLAYER-MONITOR JOYCHECK CALL, PILOTC CALL, RET,
12!-->
13:
14:
15:
```

BLK= 4

```
0:( PLAYER HOSTAGE INTERFACE JUNK )
1:F= DISH
2:SUBR dishos <ASSEMBLE O HOSTAB H LXI,
3:LABEL DISH M E MOV, H INX, M D MOV, H INX, D A MOV, E ORA, RZ,
4:IXCHG, HOSSV B LXI, B DAD, M A MOV, HSATP CPI, O=, IF,
5:HSFREE M MVI, MYFLAG HOSSV - B LXI, B DAD, 1 M MVI, THEN,
6:IXCHG, DISH JMPR, ASSEMBLE>
7:CODE DISHOS B PUSH, dishos CALL, B POP, NEXT
8:CODE HALTER HALTNOW CALL, NEXT
9!-->
10:
11:
12:
13:
14:
15:
```

BLK= 5

```
0:( CHECK VECTOR FOR INTERCEPT WITH OTHER VECTORS )
1:( ROUTINE TO FIND INTERCEPTORS, IF ANY )
2:( ENTRY: BC= NEARNESS X AND Y, HL= CHECKLIST ADDR )
3:( IY= SUBJECT VECTOR )
4:( RETURNS Z= NOFIND NZ= FIND, IX= FOUND THANG )
5:F= C:UH
6:SUBR C:U:H <ASSEMBLE
7:LABEL C:UH
8:M E MOV, H INX, M D MOV, H INX, D A MOV, E ORA,
9:RZ, D PUSH, X POPX,
10:HOSSV X A LDX, HSFREE CPI, O=, IF,
11:PROXIMITY-CHECK CALL, RNZ, THEN, C:UH JMPR,
12:ASSEMBLE>
13!-->
14:
15:
```

FILE = P

BLK= 6

```
0|( CHECK PLAYER INTERCEPT WITH OTHER VECTORS )
1|O C= EATEN 1 C= EATHOST
2|DATA CHECKLIST MONV1 , MONV2 , MONV3 , MONV4 , 0 ,
3|HEX 0202 DECIMAL C= XYBOUNDS
4|( PLAYERS INTERCEPT CHECKER, RUNS AS HOOK )
5|SUBR PLAYER:INTERCEPT:CHECK FREEZE? CALL, RNZ, EXX,
6|CHECKLIST H LXI, XYBOUNDS B LXI, CHECK:VECTOR:LIST CALL,
7|O<>, IF, 1 A MVI, PLAYERDEAD STA, FREEZE CALL,
8|EATEN FLAGCODE X MVIX, A MYFLAG X STX, ( SET EATEN FLAG )
9|3 A MVI, 4 OUT, ELSE,
10|( ANY HOSTAGE ABOUT? )
11|O HOSTAB H LXI, XYBOUNDS B LXI, C:U:H CALL,
12|O<>, IF, 1 MYFLAG X MVIX, HSATP HOSSV X MVIX, THEN,
13|A XRA, THEN, 4 OUT,
14|EXX, RET,
15|-->
```

BLK= 7

```
0|( CHECK VMAX SWITCH )
1|HEX
2|CODE VMAX? 0 H LXI, 12 IN, 5 A BIT, 0=, IF, H INX, THEN,
3|H PUSH, NEXT
4|
5|CODE SETVEL EXX, H POP, Y PUSHX, vaddr LIYD,
6|L DELTADIST Y STX, H DELTADIST 1+ Y STX, PLAYERVELO SHLD,
7|Y POPX, EXX, NEXT
8|DECIMAL -->
9|
10|
11|
12|
13|
14|
15|
```

BLK= 8

```
0|( EXPLORE-MAZE )
1|HEX 200 C= PSPDH 180 C= PSPDM 100 C= PSPDL DECIMAL
2|: EXPLORE-MAZE ;TASK: DI
3|H-P-D DISPF VB! ESTPOS
4|PLAYERANIM ANIM! XOR-ON 1STWRITE PRIBM TIMEBMAX! BEGIN DI
5|PUSH:CCR TEST:GROTTO:DRAWN IF GET:JOYCODE ELSE 0 THEN
6|IF PUSH:CCR MPLO NODE^ SWAP ADJ-SCAN
7|DUP NOWD VB@ COM 7 AND = IF HALTER THEN DUP NOWD VB!
8|DUP 32 * VANGLE VB!
9|-->
10|
11|
12|
13|
14|
15|
```

EEEE
EEEB

FILE = P

BLK= 9

```
0:( MORE PLAYER STUFF )
1:PUSH:CCR ROT TEST:REL
2:IF ZEROTIMEB
3:PUSH:CCR NOWD VB@
4:TEST:DRAWN IF
5:VMAX? IF PSPDH ELSE PSPDM
6:THEN ELSE PSPDL THEN SETVEL
7:OBJECT-MOVING BONE
8:DEPART:NODE
9:PUSH:CCR NOWD VB@ TEST:DRAWN NOT IF ( MUNCH-S )
10:100 INCSCORE REVV HEADLIGHT:REVEAL SYNC THEN
11!-->
12!
13!
14!
15!
```

BLK= 10

```
0:( EXPLORE-MAZE )
1:ELSE 0 SETVEL 3 TIMER!--ON
2:THEN ELSE 0 SETVEL 3 TIMER!--ON THEN
3:PLAYER:INTERCEPT:CHECK HOOK!--ON
4:PROPELTAS PLAYERDEAD FLAG!--ON DVECT-ON GO DI
5:OBJECT-MOVING BZERO
6:FLAG? IF DI ZEROTIMEB DEATHACT ANIM! BITE:DUST
7:0 SETVEL HALTER DISHOS
8:20 TIMER!--ON GO DI PLAYERANIM ANIM!
9:START-COL B@ NROWS 1- SET:NEW:MCCR ESTPOS
10:PLAYERDEAD ZERO THEN DI
11!-->
12!
13!
14!
15!
```

BLK= 11

```
0:( YET MORE PLAYER CONTROLLER )
1:DEST? IF ARRIVE:NODE PROPELTAS
2:TREASURE:CHECK THEN
3:0 END ; DECIMAL -->
4!
5!
6!
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

FILE = IP

BLK= 0

```
01( PROCESS A HOT ROD MISSLE )
11BV= HOTFLIP
21SUBR HOTROD
31 TBMISSLE TSTAT Y BITX, ( are we ready to process )
41 RZ, ( NOT A MISSLE )
51 ( A= timebase ) mastervmr CALL,
61 VBMISWRT VLOGICSTAT Y BITX, ( time to write ? )
71 VBMISWRT VLOGICSTAT Y RESX,
81 O<>, IF, TSUR Y L LDX, TSUR 1+ Y H LDX, FORKETH CALL,
91THEN, RET,
101-->
11!
12!
13!
14!
15!
```

BLK= 1

```
01<STKH
11SUBR MIS-INT ( missile interrupt test )
21 PSW PUSH, B PUSH, D PUSH, H PUSH, EXX, EXAF,
31 PSW PUSH, B PUSH, D PUSH, H PUSH, Y PUSHX, X PUSHX,
41( 12 IN; CMA, 1F ANI,
51JOYLAST H LXI, M CMP, A M MOV, O<>, IF, 1F A MVI, THEN,
61JOYCODE STA, ) ( HOT ROD THE PLAYERS VECTOR )
71HOTFLIP H LXI, M A MOV, A INR, 1 ANI, A M MOV,
81O=, IF, PLYRV Y LXIX, PLAYER-MONITOR CALL,
91ELSE, REVV Y LXIX, THEN, 2 A MVI,
10!HOTROD CALL,
11! SUI2-NP JMP,
12! MYPUP MYPUP MIS-INT SUI1V ! -1 HORCB OUTP ; -->
13!
14!
15!
```

FILE = M

BLK= 0

```
0:( INDEXER AND VISIBLE MONSTER WRITER )
1: INDEX:MONSTER MONVBYTES * MONV1 SWAP - ;
2:
3:SUBR VISIONWRITE ( VISIBLE MONSTER WRITER )
4: VBNOPERASE VLOGICSTAT Y BITX, 0=, IF,
5: VOPATH Y A LDX, VOPAT Y ORAX, 0<>, IF,
6: VERASE CALL, THEN, ( don't erase if no pattern )
7: ELSE, VBNOPERASE VLOGICSTAT Y RESX, THEN,
8: VBNOWRITE VLOGICSTAT Y BITX, 0=, IF, INTCPT IN, VWRITE CALL,
9: TBINTCPT-CHK TVMROPT Y BITX, 0<>, IF, INTCPT IN,
10: A ANA, 0=, IF, TBINTCPT TCHGSTAT Y SETX,
11: TBNOVECT TVMROPT Y SETX, THEN, THEN,
12: ELSE, VBNOWRITE VLOGICSTAT Y RESX, THEN,
13: transition JMP, STIO -->
14:
15:
```

BLK= 1

```
0:( MONSTER STUFF )
1:DECIMAL
2:( : NOMONST ZERODXDYAXAY 10 TIMER!-ON STAY:PUT ; )
3: BANISH:MONSTER BEGIN BEGIN NCOLS RND DUP INTC VB!
4:NOWC PLYRV OVBE - ABS 2 > END BEGIN NROWS RND DUP INTR VB!
5:NOWR PLYRV OVBE - ABS 1 > END INTC VB@ INTR VB@ NOBODY:HOME:YET?
6: END 2DROP ;
7: MONGO INTERCEPT-ON DVECT-ON
8:VISFLAG VB@ IF MYFACE V@ ANIM! VISIONWRITE ZGO DI
9:INTERCEPT? IF 0 VISFLAG VB! THEN
10:ELSE EYEBALLS-PAT ANIM! GO DI INTERCEPT? IF 1 VISFLAG VB! THEN
11:THEN COGO ;
12: FREESLAVE DI MYSLAVE V@ IF MYSLAVE V@ MYFLAG + BONE
13:( 0 MYSLAVE V@ SNATCHER + ! )
14:0 MYSLAVE V! THEN ;
15:-->
```

BLK= 2

```
0:( MORE MONSTER STUFF )
1:( COMPARE POSITION IN D AND E WITH POSITION IN VECTOR )
2:( SUBR compos D A MOV, NOWR Y CMPX, RNZ,
3:E A MOV, NOWC Y CMPX, RET, )
4:CODE CHASEPLAYER EXX, X PUSHX, Y PUSHX,
5:PLYRV X LXIX, vaddr LIYD,
6:NOWR X D LDX, NOWC X E LDX, NOWD X C LDX,
7:move:node CALL,
8:movecheck CALL, CY, IF,
9:D INTR Y STX, E INTC Y STX, THEN, EXX, Y POPX, X POPX, NEXT
10:( GO ANYWHERE I AM NOT NOW )
11: VAMOOSE BEGIN NCOLS RND INTC VB! NROWS RND INTR VB!
12:ON:TARGET? NOT END ;
13:-->
14:
15:
```

FILE = M

BLK= 3

```
0!( MONSTER TASK )
1!HEX 60 C= MONVEL
2! MONSTER-TASK ;TASK: DI
3!RETURN:INITIAL:POSITION
4!ESTPOS
5!MYFACE V@ ANIM! XOR-ON 1STWRITE BEGIN DI
6!ON:TARGET? IF SMARTS B@ RND IF CHASEPLAYER
7!ON:TARGET? IF VAMOOSE THEN ELSE VAMOOSE THEN
8! RECON SETCO COGO DI ZEROTIMEB
9!THEN FOLLOWTRACK NOWD VB!
10!MONVEL DELTADIST V! DEPART:NODE
11!( HAVE MONSTER CRAWL ABOUT )
12!BEGIN MYFLAG V^ FLAG!-ON
13! MONGO SETCO COGO DI
14!-->
15!
```

BLK= 4

```
0!( BANISHMENT STUFF )
1!FLAG? IF 0 DELTADIST V!
2!BANISH:MONSTER INTC VB@ BANC B!
3!INTR VB@ BANR B!
4! RECON SETCO COGO DI
5!0 MYFLAG VB! FLAG-OFF
6!( WANDER BACK TO WHERE MONSTER LAST CAME FROM )
7!BEGIN ESTPOS ZEROTIMEB
8!ON:TARGET? NOT IF FOLLOWTRACK NOWD VB!
9!DEPART:NODE EXITVEL DELTADIST V!
10!BEGIN MONGO SETCO COGO DEST? END ARRIVE:NODE 0
11!ELSE 1 THEN END
12!FREESLAVE
13!UNFREEZE 1 ELSE 0 DEST? IF ARRIVE:NODE DROP 1 THEN THEN
14!-->
15!
```

BLK= 5

```
0!( LAST BIT OF MONSTER STUFF )
1!END 0 END ;
2!DECIMAL -->
3!
4!
5!
6!
7!
8!
9!
10!
11!
12!
13!
14!
15!
```

MARIE

FILE = M

BLK= 6

0:(MONSTER MASH)

1:BTABLE MRTBL 0 B, 0 B, 2 B, 2 B,

2:BTABLE MCTBL 0 B, NCOLS 1- B, 0 B, NCOLS 1- B,

3: MONSTERMASH MONSTERCOUNT @ 0 DO I MCTBL B@ I MRTBL B@

4:I INDEX:MONSTER SET:INITIAL:MCCR THESPDR

5:I INDEX:MONSTER MYFACE OV! I INDEX:MONSTER MONSTER-TASK

6:LOOP ;

7:-->

8:

9:

10:

11:

12:

13:

14:

15:

FILE = VS

BLK= 0

0! (VERY STRANGE CLEAR ROUTINE)

1! HEX

2! BLUEFILL -1 4000 800 FILL :

3! DECIMAL

4! -->

5!

6!

7!

8!

9!

10!

11!

12!

13!

14!

15!



MOVE ?

FILE = E

BLK= 0

```
0:( PRE VGER ACTIVITY )
1:HEX : EXP E-C MYPUP XDI SPARKLES-OFF CLEAR:SCORES ZAP:VECT
2:8 0 DO 8 I OUTP LOOP
3:4 DUP REMAINING-LIVES ! INITIAL-LIVES !
4:GAME-OVER ZERO
5:GAME# ZERO
6:BEGIN TOTAL-PATHS ZERO
7:CHEAPRND 0 RND# !
8:MAKE:MAZE ( MAKERM ) ( MAKE DELTAS ) MD
9:SCRERASE
10:BLUEFILL
11:-->
12:
13:
14:
15:
```

BLK= 1

```
0:( MORE EXPLORE )
1:DI MYPUP ( MYINTR BAKIV ! ) ( AMUSE )
2:18 VERBL OUTP -1 HORCB OUTP
3:BREAK NOBREAK BONE ZAP:VECT
4:( TIME-BARS )
5:CLEAR:SCORE:VECTORS
6:HIDE:TREASURE JAIL:HOS
7:INPLAYERS ZERO PLAYERUP ZERO
8:REVEALED-PATHS ZERO 1 TOTAL-REVEALED-GROTTOS !
9:-->
10:
11:
12:
13:
14:
15:
```

BLK= 2

```
0:( PRE VGER ACTIVITY )
1:START-COL @ DUP PLYRV NOWC OVB!
2:REVV NOWC OVB!
3:START-ROW DUP PLYRV NOWR OVB!
4:REVV NOWR OVB! PLAYERDEAD ZERO
5:3 GAME# @ + 4 MIN MONSTERCOUNT ! ( GAME# @ DUNG# B!
6:DUNG-S ) GAME# @ 1+ 4 * 26 MIN KEY-THRESHOLD !
7:GAME# @ 1+ SMARTS B! FREEZEFLAG BZERO
8:P1SV DISPP1SCR P2SV DISPP2SCR
9:BGV INITIAL:REVEAL
10:PLYRV EXPLORE-MAZE ( JOYV JOYSTICK-MONITOR )
11:MONSTERMASH TV1 KEY-TASK
12:DISPLAY:REMAINING:LIVES 8 7 OUTP
13:-->
14:
15:
```

FILE= E

BLK= 3

01(YET MORE)

11TT GAME# 1+! NOBREAK B@ DUP 0= IF DI MYPUP 0 TVVS TVVL FILL

21BREAK TV1 END-GAME TT THEN

31(S O DO S I OUTP LOOP)

41GAME-OVER B@ OR (EMUSIC) END E-C ;

51DECIMAL -->

61

71

81

91

101

111

121

131

141

151

60
80
A0
C0
100

40

60

BLK= 4

01(THE MASTER VIDEO GAME VERBS)

11

21: VG EXP ;

31

41-->

51

61

71

81

91

101

111

121

131

141

151

OK

OK

PAGE PAGE