```
'    Professional Air Traffic Controller Simulator
'    written April, 1988 by Don Shepherd
'    Copyright 1988, Advanced Simulation Systems


'    Version 1.1  -  7/11/88
'    display the requested heading indicator arrow
'    remove "T" status from data block
'    make airport location variable via parameter file
'    speed up searches of plane.data array (use max.index)
'    provide conflict advisory service via parameter file
'    add new command "x" to disable conflict advisory for 1 plane
'    drag to reposition data blocks
'    add "cleared for the approach" command
'    suspend handoffs and close runways less often

'    Version 1.2  -  12/15/88
'    accept and initiate handoffs on a per plane basis
'    allow selective feature display on radar screen
'    allow user-selectable parameter file
'    allow restart with another parameter file option
'    add debug switch to inhibit data block flash/adjust, ca, and cz violation
checks
'    add user-definable geography display, including text
'    add "clear" key for "cleared for approach" command
'    add keypad 5 command for assigned heading
'    make airway locations variable via parameter file
'    add debug commands to get.response
'    put X in data block for newly-handed off planes
'    put random number seed value in parameter file
'    get size of Mac screen from parameter file
'    allow J4 and J13 to exit at east
'    allow n seconds to give a command
'    user inverse video instead of flashing data blocks for emphasis
'    change ascending/descending character in data block to arrows
'    react to cancel button in file dialog boxes

'    fix bug in demo mode, no handoffs if handoff zone not displayed - 9/7/89
'    fix bug in demo mode, runway planes not cleared if they originally
'      come up on final heading - 9/7/89
'    don't say handoff for new planes or exiting planes
'    fix bug in handoff barricade for south handoffs enabled - 3/24/90
'    allow computer enables in addition to computer failures

LIBRARY "atclib"
DEFINT a-z

plane.params = 23   :'number of things for each plane

DIM plane.data (23,100)  :' holds data for each plane
DIM old.plane.data (23), new.plane.data(23)   :'temp for repainting screen
DIM params(14)   :'general simulator parameters
DIM jetway.check.data (12)  :'holds heading for jetways
DIM jetway.check.name$(12)
DIM vor.ck.params (2,4)    :'check at VOR stations
DIM vor.names$(4)     :'names of VOR stations
DIM airport$(4)   :'names of the two airports
DIM airport.status(4)  :'0=open, 1=closed for 2 airports
DIM id.block.data (8,7,15)  :'params. for placement of id data block on screen
DIM coord.ck.params(4,7,15)  :'coordinate offsets for data blocks
DIM asc.symbol$(3)    :'symbols for ascent/descent
DIM hdg.direction(5,360)    :'heading direction parameters
DIM carrier$ (12)              :' 13 airlines
DIM speedfact (70)        :' move x,y based on speed
DIM runway.ck.params (2,4)    :'for checking at runways
DIM carrier.names$(12)        :'airline names, full text
DIM carrier.size (12)   :'size of carrier, small or large
DIM xy.ind (2,7) :'direction indicators
DIM depart.msg$(4) :'signoff messages
DIM warn.test (31,31) :'test if in warning area for conflict alerts
DIM req.hdgs (13)    :'counts of requested headings of jetways and runways
DIM rwarea (4,4)    :'check to see if on final for runway
DIM command.help$ (24)   :'text for command help screen
DIM ref.screen (2,15)   :'x/y params for displaying reference screen
DIM ref.screen$ (15)   :'text for reference screen
DIM stat.screen (3,13)   :'x/y params for displaying status screen
DIM stat.screen$ (13)   :'text for status screen
DIM stat.screen1 (2,16)
DIM stat.screen1$ (16)
DIM units$(19), tens$(9)  :'text for speaking numbers
DIM preferred.db.pos (8,8)   :'preferred place to put data block
DIM demo.messages$ (9)    :'messages during signon screen
DIM handoff.status (4)    :'handoffs OK to north, south, east, west
DIM arrow.data (6,8)      :'where to print the requested heading arrow
DIM arrow.hdg (12)        :'for airways and runways
DIM warn.plane(100)       :'1 if plane gets conflict alert/advisory
DIM dfv.table(3,4)        :'for cleared for approach data
DIM cycle.data(4,100,100) :'for conflict advisories
DIM geo.points(2,10,20)    :'user geography points
DIM num.pts(20)    :'number of points per line
DIM save.irec$(20)   :'save geo line data for writing back out
DIM geo.text.loc(2,20)   :'location of user text to display
DIM geo.text$(20)    :'text strings
DIM keypad.hdgs(9)    :'degree headings associated with keypad 5 command
DIM rectangle(3)    :'for inverse video of data blocks

mode.msg$(1) = "MIXED"
mode.msg$(2) = "ARTCC"
mode.msg$(3) = "APPROACH"
asc.symbol$(1) = CHR$(194)   :' ascending, up-arrow
asc.symbol$(2) = " "
asc.symbol$(3) = CHR$(160)   :'descending, down-arrow

DATA 0,180,90,270,45,225,135,315,0,180,270,90
DATA J36,J18,J09,J27,J04,J22,J13,J31,R36,R18,R27,R09
FOR i = 1 TO 12: READ jetway.check.data(i): NEXT
FOR i = 1 TO 12: READ jetway.check.name$(i): NEXT

DATA 1,5,3,7,2,6,4,8,1,5,7,3
FOR i = 1 TO 12: READ arrow.hdg(i): NEXT
```

```
vor.names$(1) = "V36"
vor.names$(2) = "V18"
vor.names$(3) = "V27"
vor.names$(4) = "V09"

depart.msg$(1) = "  "
depart.msg$(2) = ", good day sir. "
depart.msg$(3) = ", you all have a nice day. "
depart.msg$(4) = ", we'll see ya later. "

demo.messages$(0) = "delta 214, you're cleared for taikoff, runway 27"
demo.messages$(1) = "pan am 411, deescend and maintain 7 thousand feet"
demo.messages$(2) = "trans world 96, turn right to heading 270"
demo.messages$(3) = "eastern 249, clear to land, runway eighteen"
demo.messages$(4) = "american 307, clime to fourteen thousand feet"
demo.messages$(5) = "united 588, turn left to heading 180"
demo.messages$(6) = "piedmont 421, clear to land, runway 27"
demo.messages$(7) = "delta 801, cleared for taikoff, runway twenty two"
demo.messages$(8) = "air force 1, immeediat right turn to heading 135"
demo.messages$(9) = "U S Air 126, reduce speed to two hundred and fifty knots"

hand.msg1$ = " are overloaded.  please hold or divert all outbound flights until
further noughtis"
hand.msg2$ = " you may now releece outbound flights to "
close.msg1$ = " is closed.  please hold or divert all inbound flights until
further noughtis"
close.msg2$ = " has reopened.  you may resume landings there now"
direction$(1) = " north sectors "
direction$(2) = " south sectors "
direction$(3) = " east sectors "
direction$(4) = " west sectors "

DATA 1,2,0,3,7,4,6,5
DATA 2,1,3,0,4,7,5,6
DATA 3,4,2,5,1,6,0,7
DATA 4,5,3,6,2,7,1,0
DATA 5,4,6,3,7,2,0,1
DATA 6,5,7,4,0,3,1,2
DATA 7,6,0,5,1,4,2,3
DATA 0,1,7,2,6,3,5,4
FOR i = 1 TO 8
  FOR j = 1 TO 8
    READ preferred.db.pos(j,i)
  NEXT
NEXT

DATA "A     15-150   SET ALTITUDE FROM 1500 FEET TO 15000 FEET FOR SMALL PLANES"
DATA "      40-400   SET ALTITUDE FROM 4000 FEET TO 40000 FEET FOR LARGE PLANES"
DATA "      10       SET ALTITUDE TO 1000 FEET FOR RUNWAY BOUND PLANES"
DATA "B     SAME AS  SET ASSIGNED ALTITUDE AND SPEED TO THIS VALUE"
DATA "      ALT&SPD  THIS IS TYPICALLY USED FOR LANDING PLANES"
DATA "C     NONE     CLEARED FOR THE APPROACH"
DATA "D     9,27,18,36  DIVERT PLANE TO A RUNWAY"
DATA "H     0-359    SET ASSIGNED HEADING TO THIS DEGREE SETTING"
DATA "P     NONE OR  PROJECT FUTURE FLIGHT POSITION"
DATA "      1-15     SHOW FLIGHT POSITION IN THIS MANY MINUTES"
DATA "S     10-20    SET SPEED FROM 100 KNOTS TO 200 KNOTS FOR SMALL PLANES"
DATA "      20-70    SET SPEED FROM 200 KNOTS TO 700 KNOTS FOR LARGE PLANES"
DATA "      10       SET SPEED TO 100 KNOTS FOR RUNWAY BOUND PLANES"
DATA "V     9,27,18,36   HEAD FOR THE VOR (NORMALLY USED FOR LANDING PLANES)"
DATA "W     NONE OR  TOGGLE WARNING AREA DISPLAY"
DATA "      1-50     DISPLAY WARNING AREA FOR THIS MANY CYCLES"
DATA "X     NONE     TOGGLE CONFLICT ADVISORY ENABLE STATUS"
DATA ".     NONE     SET ASSIGNED ALTITUDE, SPEED, AND HEADING = REQUESTED
VALUES"
DATA "1-9   NONE     REPOSITION DATA BLOCK (LIKE NUMERIC KEYPAD LAYOUT)"
DATA "0     NONE     ACCEPT OR INITIATE A HANDOFF"
DATA "+-    1-15     INCREASE/DECREASE LENGTH OF DATA BLOCK LEADER LINE"
DATA "*     NONE     SET LEADER LINE TO NORMAL LENGTH"
DATA "=     NONE OR  HOLD TOGGLE"
DATA "      1-50     HOLD FOR THIS MANY CYCLES"
FOR i = 1 TO 24: READ command.help$(i): NEXT

DATA 1,14,"J31"
DATA 1,41,"J04"
DATA 1,59,"J36"
DATA 15,1,"J27"
DATA 15,82,"J09"
DATA 23,1,"J22"
DATA 28,59,"J18"
DATA 28,71,"J13"
DATA 16,65,"CONTROL ZONE"
DATA 19,22,"CONTROL ZONE"
DATA 21,18,"V09   R09   R27   V27"
DATA 4,66,"V18"
DATA 7,66,"R18"
DATA 10,66,"R36"
DATA 13,66,"V36"
FOR i = 1 TO 15: READ ref.screen(1,i), ref.screen(2,i), ref.screen$(i): NEXT

DATA 19,44,"J36 - ",1
DATA 19,56,"J04 - ",5
DATA 19,68,"R36 - ",9
DATA 20,44,"J18 - ",2
DATA 20,56,"J22 - ",6
DATA 20,68,"R18 - ",10
DATA 21,44,"J09 - ",3
DATA 21,56,"J13 - ",7
DATA 21,68,"R09 - ",12
DATA 22,44,"J27 - ",4
DATA 22,56,"J31 - ",8
DATA 22,68,"R27 - ",11
DATA 24,44,"DEGREE - ",13
FOR i = 1 TO 13: READ stat.screen(1,i), stat.screen(2,i), stat.screen$(i),
stat.screen(3,i): NEXT

DATA 1,68,"COMMAND SUMMARY:"
DATA 2,68,"A - ALTITUDE"
DATA 3,68,"B - ALT & SPEED"
DATA 4,68,"C - CLRD FOR APPR."
DATA 5,68,"D - DIVERT/RUNWAY"
DATA 6,68,"H - HEADING"
DATA 7,68,"P - PROJECT PATH"
DATA 8,68,"S - SPEED"
DATA 9,68,"V - VOR HEADING"
DATA 10,68,"W - WARNING AREA"
DATA 11,68,"X - STOP ADVISORY"
DATA 12,68,". - ASG = REQ"
DATA 13,68,"1-9 - DATA BLOCK"
DATA 14,68,"0 - HANDOFF"
DATA 15,68,"+-* - LEADER LINE"
DATA 16,68,"= - HOLD"
FOR i = 1 TO 16: READ stat.screen1(1,i), stat.screen1(2,i), stat.screen1$(i):
NEXT

DATA one, two, three, four, five, six, seven, eight, nine, ten
DATA eleven, twelve, thir teen, four teen, fif teen
DATA six teen, seven teen, eight teen, nighn teen
DATA ten, twenty, thirty, forty, fifty, sixty, seventy, eighty, nighty
FOR i = 1 TO 19: READ units$(i): NEXT
FOR i = 1 TO 9: READ tens$(i): NEXT
FOR i = 1 TO 19: units$(i)= " "+units$(i)+" ": NEXT
FOR i = 1 TO 9: tens$(i)=" "+tens$(i)+" ": NEXT

idblkpos(1) = 2
idblkpos(2) = 1
idblkpos(3) = 0
idblkpos(4) = 3
idblkpos(6) = 7
idblkpos(7) = 4
idblkpos(8) = 5
idblkpos(9) = 6

DATA 1,1,0,1,-1,1,-1,0,-1,-1,0,-1,1,-1,1,0

FOR x = 0 TO 7
  FOR y = 1 TO 2
    READ xy.ind(y,x) :'xy offset factors for data block positions
  NEXT
NEXT

DATA 3,3,9,9,12,12,12,21
DATA 0,3,0,9,-17,19,-17,28
DATA -3,3,-9,9,-45,12,-45,21
DATA -3,0,-9,0,-45,0,-45,9
DATA -3,-3,-9,-9,-45,-12,-45,-3
DATA 0,-3,0,-9,-17,-19,-17,-10
DATA 3,-3,9,-9,12,-12,12,-3
DATA 3,0,9,0,12,0,12,9

FOR i = 0 TO 7
    FOR j = 1 TO 8
        READ id.block.data(j,i,0)
    NEXT
```

```
NEXT

FOR h = 1 TO 15
  FOR i = 0 TO 7

id.block.data(1,i,h)=id.block.data(1,i,0):id.block.data(2,i,h)=id.block.data(2,i
,0)

id.block.data(3,i,h)=id.block.data(3,i,0)+(h*6*xy.ind(1,i)):id.block.data(4,i,h)
=id.block.data(4,i,0)+(h*6*xy.ind(2,i))

id.block.data(5,i,h)=id.block.data(5,i,0)+(h*6*xy.ind(1,i)):id.block.data(6,i,h)
=id.block.data(6,i,0)+(h*6*xy.ind(2,i))

id.block.data(7,i,h)=id.block.data(7,i,0)+(h*6*xy.ind(1,i)):id.block.data(8,i,h)
=id.block.data(8,i,0)+(h*6*xy.ind(2,i))
  NEXT
NEXT

DATA 12,3,48,21
DATA -17,10,19,28
DATA -45,3,-9,21
DATA -45,-9,-9,9
DATA -45,-21,-9,-3
DATA -17,-28,19,-10
DATA 12,-21,48,-3
DATA 12,-9,48,9

FOR i = 0 TO 7   :'coordinates to see if you clicked on a data block
    FOR j = 1 TO 4
        READ coord.ck.params(j,i,0)
    NEXT
NEXT

FOR h = 1 TO 15
  FOR i = 0 TO 7
    coord.ck.params(1,i,h) = coord.ck.params(1,i,0)+(h*6*xy.ind(1,i))
    coord.ck.params(2,i,h) = coord.ck.params(2,i,0)+(h*6*xy.ind(2,i))
    coord.ck.params(3,i,h) = coord.ck.params(3,i,0)+(h*6*xy.ind(1,i))
    coord.ck.params(4,i,h) = coord.ck.params(4,i,0)+(h*6*xy.ind(2,i))
  NEXT
NEXT

DATA 1,25,2,21,3,18,4,16,5,15,6,13,7,12,8,10,9,9

FOR i = 1 TO 9
  READ x,y
  FOR j = x TO y
    warn.test(32-j,32-i) = 1
    warn.test(32-i,32-j) = 1
  NEXT
NEXT

DATA 0,-6,-1,-5,1,-5
DATA 5,-5,4,-5,5,-4
DATA 6,0,5,-1,5,1
DATA 5,5,5,4,4,5
DATA 0,6,-1,5,1,5
DATA -5,5,-4,5,-5,4
DATA -6,0,-5,-1,-5,1
DATA -5,-5,-5,-4,-4,-5

FOR i = 1 TO 8   :'where to print requested heading arrow
  FOR j = 1 TO 6
    READ arrow.data(j,i)
  NEXT
NEXT

DATA 5,20,20,30,30,30,70,40,40,120,50,50
FOR i = 1 TO 4: FOR j = 1 TO 3: READ dfv.table(j,i): NEXT: NEXT

DATA 225,180,135,270,0,90,315,0,45
FOR i = 1 TO 9: READ keypad.hdgs(i): NEXT

FOR i = 0 TO 70    :'speed factors to adjust x,y each cycle
    speedfact(i) = INT(i/10)+1
NEXT

FOR i = 338 TO 360
    hdg.direction(1,i) = 1
    hdg.direction(2,i) = 0
    hdg.direction(3,i) = -3
    hdg.direction(4,i) = 0
    hdg.direction(5,i) = -1
NEXT
FOR i = 0 TO 22
    hdg.direction(1,i) = 1
    hdg.direction(2,i) = 0
    hdg.direction(3,i) = -3
    hdg.direction(4,i) = 0
    hdg.direction(5,i) = -1
NEXT
FOR i = 23 TO 67
    hdg.direction(1,i) = 2
    hdg.direction(2,i) = 3
    hdg.direction(3,i) = -3
    hdg.direction(4,i) = 1
    hdg.direction(5,i) = -1
    hdg.direction(1,i+45) = 3
    hdg.direction(2,i+45) = 3
    hdg.direction(3,i+45) = 0
    hdg.direction(4,i+45) = 1
    hdg.direction(5,i+45) = 0
    hdg.direction(1,i+90) = 4
    hdg.direction(2,i+90) = 3
    hdg.direction(3,i+90) = 3
    hdg.direction(4,i+90) = 1
    hdg.direction(5,i+90) = 1
    hdg.direction(1,i+135) = 5
    hdg.direction(2,i+135) = 0
    hdg.direction(3,i+135) = 3
    hdg.direction(4,i+135) = 0
    hdg.direction(5,i+135) = 1
    hdg.direction(1,i+180) = 6
    hdg.direction(2,i+180) = -3
    hdg.direction(3,i+180) = 3
    hdg.direction(4,i+180) = -1
    hdg.direction(5,i+180) = 1
    hdg.direction(1,i+225) = 7
    hdg.direction(2,i+225) = -3
    hdg.direction(3,i+225) = 0
    hdg.direction(4,i+225) = -1
    hdg.direction(5,i+225) = 0
    hdg.direction(1,i+270) = 8
    hdg.direction(2,i+270) = -3
    hdg.direction(3,i+270) = -3
    hdg.direction(4,i+270) = -1
    hdg.direction(5,i+270) = -1
NEXT

start.all.over:    :'if you restart, it comes back here

debug.switch = 1    :'1=normal, -1=debug mode

GOSUB read.params   :'read the simulation parameters from file "ATC params"
GOSUB airport.setup   :'setup airport location parameters

'  setup all menus

MENU 1,0,1,"Control"
MENU 1,1,1,"Quit"
MENU 1,2,1,"Play again"
MENU 1,3,1,"Set options"
MENU 1,4,1,"Start all over"
MENU 2,0,1,"Display"
MENU 2,1,1,"Radar screen"
MENU 2,2,1,"Status screen"
MENU 2,3,1,"Score screen"
MENU 2,4,1,"Reference screen"
MENU 2,5,1,"Command description"
MENU 3,0,1,"Radar"
MENU 3,1,dswt.fdb+1,"full data blocks"
MENU 3,2,dswt.a+1,"airways"
MENU 3,3,dswt.r+1,"runways"
MENU 3,4,dswt.v+1,"VORs"
MENU 3,5,dswt.cz+1,"control zones"
MENU 3,6,dswt.hz+1,"handoff zones"
MENU 3,7,dswt.g+1,"geography"
MENU 3,8,dswt.rb+1,"runway barricades"
MENU 3,9,dswt.hb+1,"handoff barricades"
IF params(1) = 1 THEN MENU 4,0,1,"Demonstration Mode" ELSE MENU 4,0,0,""

CmdKey! 1,1,"Q"
CmdKey! 1,2,"P"
CmdKey! 1,3,"O"
```

```
CmdKey! 2,1,"R"
CmdKey! 2,2,"S"
CmdKey! 2,3,"A"
CmdKey! 2,4,"Z"
CmdKey! 2,5,"C"
ChangeCursor! 2   :'make it a crosshairs

'  initialize MacinTalk

speechhand! = 0!
speecherr% = 0
phonh! = 0
speechon! "",speechhand!,speecherr%
speechrate! speechhand!,voice.speed(params(8))

GOSUB startup

run.again:  ' if you run again, it comes back to here

REM zero out the plane data array
FOR i = 0 TO max.planes-1
  FOR j = 0 TO plane.params
    plane.data(j,i) = 0
  NEXT
NEXT

handoff.status(1) = 0  :'normal
handoff.status(2) = 0
handoff.status(3) = 0
handoff.status(4) = 0
airport.status(1) = 0  :'open
airport.status(2) = 0
airport.status(3) = 0
airport.status(4) = 0


REM these counters are displayed when you quit the simulation
start.date$ = DATE$
start.time$ = TIME$
stop.time$ = " "  :'will be set when you quit
max.planes.handled = 0  :'set to max. planes active at one time
planes.exited.successfully = 0  :'at proper alt/speed/heading
planes.landed.successfully = 0
planes.exited.bad.alt = 0
planes.exited.bad.spd = 0
planes.exited.bad.hdg = 0
planes.exited.bad.sector = 0
planes.crashed.total = 0
planes.proximity.warnings = 0
planes.tca.violations = 0
cf.cycles = 0
planes.activated = 0

ON MENU GOSUB menuproc   :'go here on a menu selection
ON MOUSE GOSUB mouseproc   :'go here on a mouse press
ON DIALOG GOSUB dialogproc   :'go here on a dialog box response

IF options = 1 THEN GOSUB display.options: options = 0

IF params(12) = 1 THEN params(11) = INT(200*RND)   :'planes/shift
IF params(10) = 1 THEN params(9) = INT(15*RND)+1   :'planes to start with
IF params(9) > params(11) THEN params(9) = params(11)   :'don't start with more
than in shift

REM initialize plane.data array
FOR j = 0 TO params(9)-1
    GOSUB make.new.plane
NEXT
max.index = params(9)-1  :'largest referenced position in plane.data

'  setup main radar window

WINDOW 1,,(2,22)-(mac.screen.width-2,mac.screen.height-4),3
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type

GOSUB paint.with.clear    :'paint screen the first time

begin:
    ChangeCursor! 2   :'make it a crosshairs
    GOSUB delayNsecs                    :'delay for N seconds
    IF rerun.sw = 1 THEN rerun.sw = 0: GOTO run.again    :'rerun simulation
    IF rerun.sw1 = 1 THEN rerun.sw1 = 0: GOTO start.all.over   :'all over again
    IF params(1) = 1 THEN GOSUB demo.proc   :'issue commands automatically
    GOSUB paint.update.screen        :'update plane data base and paint current
screen
    IF rerun.sw = 1 THEN rerun.sw = 0: GOTO run.again    :'rerun simulation
    IF rerun.sw1 = 1 THEN rerun.sw1 = 0: GOTO start.all.over   :'all over again
    GOSUB dothechecks                   :'do all checks and special requests
    GOTO begin

REM  **********************************************
REM  all subroutines are listed here
REM  in alphabetical order
REM  **********************************************

'adjust.cfa.alt.speed:    'adjust alt and speed if cleared for approach
'advisory.check:  'check for long-term conflict alerts, if enabled
'airport.setup:   'setup airport location parameters
'at.the.jetway.check:   'give advisory when a jetway-bound plane reaches the
jetway
'build.cmd.prompt:   'construct the prompt for giving a plane command
'build.options.screen:   'create all edit fields and buttons
'check.overlap:   'check for overlapping data blocks
'construct.response:   'build op response, but only allow n secs for input
'ctalk:   'speak the controllers commands to the planes
'data.block.adjust:   'check for overlapping data blocks and adjust
'delayNsecs:    'delay for N seconds, allowing mouse clicks
'demo.proc:   'issue commands automatically in demo mode
'dialogproc:   'handle dialog entries in the options screen
```

```
'display.command.help:   'display the command help screen
'display.options:   'display the options screen
'display.ref.screen:   'display the reference screen, with all features labelled
'display.score:   'display the score screen
'display.status:   'display the status screen
'dothechecks:   'perform these checks during every update cycle
'drag.db:    'reposition the data block by dragging it
'draw.airways:   'print airways, runways, VORs, and control zones on the screen
'draw.airways.ref:   'print airways for reference screen only
'draw.geography:   'draw user-defined geography features
'draw.handoff.zone:   'draw the dotted lines for the handoff zone
'figure.score:   'calculate the current score, put in ts
'gen.comp.failure:   'generate a computer failure
'get.response:   'get a command from the operator
'handle.button:   'user has pressed a button in the options screen
'handle.editfield:   'user has entered an edit field in the options screen
'handle.handoff:   'see if you want to close an airport or handoffs
'inverse.video:   'reverse the color of data blocks, for emphasis
'landing.check:   'see if a plane destined for a runway has landed, and give msg
'make.new.plane:    'to create everything for a new plane
'menuproc:   'go here when a menu choice is made
'mouseproc:   'go here when a mouse press is made
'move.db:   'move the data block based upon a mouse drag
'paint.it.black:   'draw one plane on screen
'paint.it.white:   'erase one plane from screen
'paint.update.screen:   'paint plane white, update vals, paint black
'paint.with.clear:    'clear the screen and repaint everything
'paint.wo.clear:   'paint one plane on the screen (index k)
'parse.command:   'interpret multiple commands on one line
'proximity.check:    'issue alert if planes are too close
'read.params:   'read the simulation parameters from the file "ATC params"
'redraw.one.plane:    'redraw one plane on the screen
'release.from.hold:   'see if it is time to stop holding
'save.options:   'writes the current options to the file "ATC params"
'saynum:   'convert a number (sayn!) into speech (sayn$)
'selective.display:    'handle user-selectable displays on radar screen
'set.vor.hdg:   'if the plane is on a heading to a VOR, set current heading
'set.cfa.hdg:   'if the plane is cleared for approach, check for arrival at VOR
'speakit:   'pronounce the string in ts$
'speak.text:   'test pronounces center name, airports, or airlines
'speak.invalid.msg:   'bad user input
'specl.requests:   'generate special requests every so often
'specl.req.mistake:   'generate a mistake for a plane
'specl.req.newplane:   'generate all parameters for a new plane
'specl.req.oldplane:   'generate a change in alt/spd/hdg for an old plane
'startup:   'display the startup screen
'startup.dialog:   'handle initial startup screen
'stop.Nsec.delay:   'stop the delay cycle
'tca.violation.check:    'see if a plane violates the control zone of an airport
'update.airline.size:   'change the airline size from dialog screen
'update.counts:   'update random switches on dialog display
'update.current.values:    'update current alt/speed/heading
'update.levels:   'update parameter levels on dialog display
'update.modes:   'update modes on dialog display
'were.gone:    'plane has left the screen, tally and say goodbye
```

```
'write.modes1:   'rewrite mode settings on dialog display
'write.modes2:   'rewrite level settings on dialog display
'write.modes3:   'rewrite count settings on dialog display

REM  ***********************************************
adjust.cfa.alt.speed:    'adjust alt and speed if cleared for approach

FOR i = 0 TO max.index
  IF plane.data(0,i) <> 1 GOTO adj.cfa.exit    :'not active
  IF plane.data(22,i) <> 1 GOTO adj.cfa.exit    :'not cleared for approach
  IF plane.data(12,i) >= 0 GOTO adj.cfa.exit    :'not on VOR hdg anymore
(holding or on final)
  xdist.from.vor = ABS(plane.data(1,i)-vor.ck.params(1,ABS(plane.data(12,i))))
  ydist.from.vor = ABS(plane.data(2,i)-vor.ck.params(2,ABS(plane.data(12,i))))
  IF xdist.from.vor > ydist.from.vor THEN dfv = xdist.from.vor ELSE dfv =
ydist.from.vor
  a=20: s=20
  FOR u= 1 TO 4
    IF dfv >= dfv.table(1,u) THEN a=dfv.table(2,u): s=dfv.table(3,u)
  NEXT
  IF carrier.size(plane.data(3,i)) = 0 AND s > 20 THEN s = 20
  IF plane.data(10,i) = a AND plane.data(11,i) = s GOTO adj.cfa.exit
  k = i: GOSUB paint.it.white
  plane.data(10,i) = a
  plane.data(11,i) = s
  k = i: GOSUB paint.it.black

  adj.cfa.exit:
NEXT

RETURN

REM  ***********************************************
advisory.check:  'check for long-term conflict alerts, if enabled

IF params(14) <> 1 THEN RETURN  :'function is disabled by user

'calculate array cycle.data with projected position, altitude, and speed

FOR i = 0 TO max.index
  cycle.data(1,i,0) = plane.data(1,i)  :'x coordinate
  cycle.data(2,i,0) = plane.data(2,i)  :'y coordinate
  cycle.data(3,i,0) = plane.data(5,i)  :'current altitude
  cycle.data(4,i,0) = plane.data(6,i)  :'current speed
NEXT

FOR i = 1 TO adv.cycles   :'calculate rest of array
  FOR j = 0 TO max.index
    IF plane.data(0,j) <> 1 GOTO adv.exit  :'inactive or unident, ignore
    IF warn.plane(j) <> 0 GOTO adv.exit
    IF plane.data(21,j) = 1 GOTO adv.exit  :'disabled for this plane
    hdg = plane.data(7,j)
    n = speedfact(cycle.data(4,j,i-1))
    IF plane.data(17,j) <> 0 THEN n = 0  :'holding so don't adjust x/y
    cycle.data(1,j,i) = cycle.data(1,j,i-1) + (n*hdg.direction(4,hdg))
    cycle.data(2,j,i) = cycle.data(2,j,i-1) + (n*hdg.direction(5,hdg))
    IF cycle.data(3,j,i-1) = plane.data(10,j) THEN cycle.data(3,j,i) =
cycle.data(3,j,i-1): GOTO adv.xy
    IF cycle.data(3,j,i-1) > plane.data(10,j) THEN cycle.data(3,j,i) =
cycle.data(3,j,i-1) - 1 ELSE cycle.data(3,j,i) = cycle.data(3,j,i-1) + 1
    adv.xy:
    IF cycle.data(4,j,i-1) = plane.data(11,j) THEN cycle.data(4,j,i) =
cycle.data(4,j,i-1): GOTO adv.xz
    IF cycle.data(4,j,i-1) > plane.data(11,j) THEN cycle.data(4,j,i) =
cycle.data(4,j,i-1) - 1 ELSE cycle.data(4,j,i) = cycle.data(4,j,i-1) + 1
    adv.xz:

    adv.exit:
  NEXT
NEXT

'now check for conflicts at any cycle
FOR i = 1 TO adv.cycles

  FOR j = 0 TO max.index - 1

    IF plane.data(0,j) <> 1 GOTO adv1.exit  :'inactive or unident, ignore
    IF cycle.data(1,j,i) < 0 OR cycle.data(1,j,i) > mac.screen.width-4 GOTO
adv1.exit
    IF cycle.data(2,j,i) < 0 OR cycle.data(2,j,i) > mac.screen.height-26 GOTO
adv1.exit
    IF warn.plane(j) <> 0 GOTO adv1.exit
    IF plane.data(21,j) = 1 GOTO adv1.exit  :'disabled for this plane

    FOR k = j+1 TO max.index

      IF plane.data(0,k) <> 1 GOTO adv2.exit  :'inactive or unident, ignore
      IF cycle.data(1,k,i) < 0 OR cycle.data(1,k,i) > mac.screen.width-4 GOTO
adv2.exit
      IF cycle.data(2,k,i) < 0 OR cycle.data(2,k,i) > mac.screen.height-26 GOTO
adv2.exit
      IF warn.plane(k) <> 0 GOTO adv2.exit
      IF plane.data(21,k) = 1 GOTO adv2.exit  :'disabled for this plane

      'check for conflict between planes j and k at cycle i

      pck.x1 = cycle.data(1,j,i): pck.y1 = cycle.data(2,j,i): pck.alt1 =
cycle.data(3,j,i)
      pck.x2 = cycle.data(1,k,i): pck.y2 = cycle.data(2,k,i): pck.alt2 =
cycle.data(3,k,i)
      xdif = ABS(pck.x1 - pck.x2): ydif = ABS(pck.y1 - pck.y2): altdif =
ABS(pck.alt1 - pck.alt2)
      IF altdif >= 10 GOTO adv2.exit  :'alt separation is more than 1000 feet
      IF xdif > 30 OR ydif > 30 GOTO adv2.exit  :'they aren't that close
      IF warn.test(xdif+1,ydif+1) = 1 GOTO adv2.exit  :'not in circle

      IF params(4) = 1 THEN BEEP
      tt1$ = carrier.names$(plane.data(3,j)) + STR$(plane.data(4,j))
      tt2$ = carrier.names$(plane.data(3,k)) + STR$(plane.data(4,k))
      IF params(3) <> 1 THEN ndx = j: GOSUB inverse.video
      IF params(3) <> 1 THEN ndx = k: GOSUB inverse.video
      tts$ = "KON FLICT AD VYZE OREE  -----  " + tt1$ + " and " + tt2$
      voice.type = -1
      GOSUB speakit
      IF params(3) <> 1 THEN ndx = k: GOSUB inverse.video
      IF params(3) <> 1 THEN ndx = j: GOSUB inverse.video
      warn.plane(j) = 1
      warn.plane(k) = 1

      adv2.exit:
    NEXT

    adv1.exit:
  NEXT

NEXT

RETURN

REM  ***********************************************
airport.setup:   'setup airport location parameters

vor.ck.params(1,1) = ap1x+1
vor.ck.params(2,1) = ap1y+70
vor.ck.params(1,2) = ap1x+1
vor.ck.params(2,2) = ap1y-40
vor.ck.params(1,3) = ap2x+70
vor.ck.params(2,3) = ap2y+1
vor.ck.params(1,4) = ap2x-40
vor.ck.params(2,4) = ap2y+1
rwarea(1,1)=ap1x+1-5: rwarea(2,1)=ap1x+1+5: rwarea(3,1)=ap1y+30-5:
rwarea(4,1)=ap1y+70+5
rwarea(1,2)=ap1x+1-5: rwarea(2,2)=ap1x+1+5: rwarea(3,2)=ap1y-40-5:
rwarea(4,2)=ap1y+5
rwarea(1,3)=ap2x+30-5: rwarea(2,3)=ap2x+70+5: rwarea(3,3)=ap2y+1-5:
rwarea(4,3)=ap2y+1+5
rwarea(1,4)=ap2x-40-5: rwarea(2,4)=ap2x+5: rwarea(3,4)=ap2y+1-5:
rwarea(4,4)=ap2y+1+5
runway.ck.params(1,1) = ap1x+1: runway.ck.params(2,1) = ap1y+30
runway.ck.params(1,2) = ap1x+1: runway.ck.params(2,2) = ap1y
runway.ck.params(1,3) = ap2x+30: runway.ck.params(2,3) = ap2y+1
runway.ck.params(1,4) = ap2x: runway.ck.params(2,4) = ap2y+1

RETURN

REM  ***********************************************
at.the.jetway.check:   'give advisory when a jetway-bound plane reaches the
jetway

REM if a plane has arrived (+-7 pixels) at the desired jetway, print a message
max.planes.counter = 0
IF params(3) = 1 THEN cf.cycles = cf.cycles + 1
FOR jck.idx = 0 TO max.index
  IF plane.data(0,jck.idx) = 1 THEN max.planes.counter = max.planes.counter + 1
  IF plane.data(0,jck.idx) = 0 GOTO at.jetway.loop.exit  :'inactive plane
  IF plane.data(1,jck.idx) < 0 OR plane.data(1,jck.idx) > mac.screen.width-4
THEN GOSUB were.gone :GOTO at.jetway.loop.exit
  IF plane.data(2,jck.idx) < 0 OR plane.data(2,jck.idx) > mac.screen.height-26
THEN GOSUB were.gone :GOTO at.jetway.loop.exit
  IF plane.data(0,jck.idx) = 2 GOTO at.jetway.loop.exit  :'unidentified plane
  IF plane.data(15,jck.idx) >= 0 OR  plane.data(15,jck.idx) <= -9 GOTO
at.jetway.loop.exit  :'not on jetway heading
  IF plane.data(12,jck.idx) = jetway.check.data(ABS(plane.data(15,jck.idx)))
GOTO at.jetway.loop.exit  :'is on the proper heading already
  cur.x.pos = plane.data(1,jck.idx)
  cur.y.pos = plane.data(2,jck.idx)
  IF (plane.data(15,jck.idx) = -1 OR plane.data(15,jck.idx) = -2) AND (cur.x.pos
>= aw3618x1-7 AND cur.x.pos <= aw3618x1+7) THEN GOTO prt.at.msg
  IF (plane.data(15,jck.idx) = -3 OR plane.data(15,jck.idx) = -4) AND (cur.y.pos
>= aw0927y1-7 AND cur.y.pos <= aw0927y1+7) THEN GOTO prt.at.msg
  IF (plane.data(15,jck.idx) = -5 OR plane.data(15,jck.idx) = -6) AND
(cur.x.pos+cur.y.pos >= aw0422y1-10 AND cur.x.pos+cur.y.pos <= aw0422y1+10) THEN
GOTO prt.at.msg
  IF (plane.data(15,jck.idx) = -7 OR plane.data(15,jck.idx) = -8) AND
(cur.x.pos-cur.y.pos >= aw1331x1-10 AND cur.x.pos-cur.y.pos <= aw1331x1+10) THEN
GOTO prt.at.msg

  GOTO at.jetway.loop.exit
  prt.at.msg:
  new.hdg = plane.data(12,jck.idx)
  tts$ = carrier.names$(plane.data(3,jck.idx))+STR$(plane.data(4,jck.idx))+" IS
AT "+ jetway.check.name$(ABS(plane.data(15,jck.idx)))
  sayn! = jetway.check.data(ABS(plane.data(15,jck.idx)))
  GOSUB saynum
  IF params(2) = 1 THEN tts$ = tts$ + " AND IS TURNING TO HEADING "+sayn$+"
DEGREES ": new.hdg=jetway.check.data(ABS(plane.data(15,jck.idx)))
  voice.type = jck.idx
  GOSUB speakit
  k=jck.idx
  GOSUB paint.it.white
  plane.data(12,jck.idx) = new.hdg
  GOSUB paint.it.black
  at.jetway.loop.exit:
NEXT
IF max.planes.counter > max.planes.handled THEN max.planes.handled =
max.planes.counter
RETURN

REM  **********************************************
build.cmd.prompt:   'construct the prompt for giving a plane command

mouseplane = mousez
x = plane.data(1,mousez): y = plane.data(2,mousez): z = plane.data(9,mousez): zz
= plane.data(20,mousez)
LINE (x+id.block.data(5,z,zz)-2,y+id.block.data(6,z,zz)-9) -
(x+id.block.data(5,z,zz)+36,y+id.block.data(6,z,zz)+10),,B
WINDOW 2,,(2,22)-(mac.screen.width-2,33),3
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
```

```
mousecarrier$ = carrier$(plane.data(3,mouseplane))
mouseflight$ =
RIGHT$(STR$(plane.data(4,mouseplane)),LEN(STR$(plane.data(4,mouseplane)))-1)
mousexcord$ =
RIGHT$(STR$(plane.data(1,mouseplane)),LEN(STR$(plane.data(1,mouseplane)))-1)
mouseycord$ =
RIGHT$(STR$(plane.data(2,mouseplane)),LEN(STR$(plane.data(2,mouseplane)))-1)
mousealt1$ =
RIGHT$(STR$(plane.data(5,mouseplane)),LEN(STR$(plane.data(5,mouseplane)))-1)
mousealt2$ =
RIGHT$(STR$(plane.data(10,mouseplane)),LEN(STR$(plane.data(10,mouseplane)))-1)
mousealt3$ =
RIGHT$(STR$(plane.data(13,mouseplane)),LEN(STR$(plane.data(13,mouseplane)))-1)
mousespeed1$ =
RIGHT$(STR$(plane.data(6,mouseplane)),LEN(STR$(plane.data(6,mouseplane)))-1)
mousespeed2$ =
RIGHT$(STR$(plane.data(11,mouseplane)),LEN(STR$(plane.data(11,mouseplane)))-1)
mousespeed3$ =
RIGHT$(STR$(plane.data(14,mouseplane)),LEN(STR$(plane.data(14,mouseplane)))-1)
mousehdg1$ =
RIGHT$(STR$(plane.data(7,mouseplane)),LEN(STR$(plane.data(7,mouseplane)))-1)
mousehdg2$ =
RIGHT$(STR$(plane.data(12,mouseplane)),LEN(STR$(plane.data(12,mouseplane)))-1)
IF plane.data(12,mouseplane) < 0 THEN mousehdg2$ =
vor.names$(ABS(plane.data(12,mouseplane)))
mousehdg3$ =
RIGHT$(STR$(plane.data(15,mouseplane)),LEN(STR$(plane.data(15,mouseplane)))-1)
IF plane.data(15,mouseplane) < 0 THEN mousehdg3$ =
jetway.check.name$(ABS(plane.data(15,mouseplane)))

REM build the prompt string for display

prompt$ = mousecarrier$ + mouseflight$ + " (" + mousexcord$ + "," + mouseycord$
+ ") (ALT "
prompt$ = prompt$ + mousealt1$ + "/" + mousealt2$ + "/" + mousealt3$ + ") (SPD "
prompt$ = prompt$ + mousespeed1$ + "/" + mousespeed2$ + "/" + mousespeed3$ + ")
(HDG "
prompt$ = prompt$ + mousehdg1$ + "/" + mousehdg2$ + "/" + mousehdg3$ + ")"
RETURN

REM  ***********************************************
build.options.screen:   'create all edit fields and buttons

edit.fld.type = 5   :'use 1 for interpreter, 5 for compiler
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
n = 14
CALL MOVETO (1,n)
DrawText! "CENTER"
EDIT FIELD 1,center.name$,(50,5)-(200,15),edit.fld.type
BUTTON 1,1,"SAY",(210,n-10)-(240,n+3),1
n=n+15
CALL MOVETO (1,n)
DrawText! "AIRPORT"
EDIT FIELD 2,airport$(1),(50,20)-(200,30),edit.fld.type
```

```
BUTTON 2,1,"SAY",(210,n-10)-(240,n+3),1
n=n+15
CALL MOVETO (1,n)
DrawText! "AIRPORT"
EDIT FIELD 3,airport$(3),(50,35)-(200,45),edit.fld.type
BUTTON 3,1,"SAY",(210,n-10)-(240,n+3),1

n=80
CALL MOVETO (1,n-10)
DrawText! "big ID    Airline name"
FOR i = 0 TO 12
  BUTTON 4+i,carrier.size(i)+1,"",(2,n-1)-(20,n+12),2
  EDIT FIELD 4+i,carrier$(i),(24,n)-(40,n+10),edit.fld.type
  EDIT FIELD 17+i,carrier.names$(i),(50,n)-(200,n+10),edit.fld.type
  BUTTON 17+i,1,"SAY",(210,n-1)-(240,n+12),1
  n=n+15
NEXT

BUTTON 30,1,"SAVE ALL OPTIONS",(10,285)-(130,310),1
BUTTON 31,1,"CONTINUE",(145,285)-(240,310),1

GOSUB write.modes1
GOSUB write.modes2
GOSUB write.modes3

RETURN

REM  ***********************************************
check.overlap:   'check for overlapping data blocks

pix = plane.data(1,i)  :'plane dot coordinates
piy = plane.data(2,i)
pjx = plane.data(1,j)
pjy = plane.data(2,j)

'  data block corner coordinates of planes i and j
pix1 = plane.data(1,i) + coord.ck.params(1,plane.data(9,i),plane.data(20,i))
piy1 = plane.data(2,i) + coord.ck.params(2,plane.data(9,i),plane.data(20,i))
pix2 = plane.data(1,i) + coord.ck.params(3,plane.data(9,i),plane.data(20,i))
piy2 = plane.data(2,i) + coord.ck.params(4,plane.data(9,i),plane.data(20,i))

pjx1 = plane.data(1,j) + coord.ck.params(1,plane.data(9,j),plane.data(20,j))
pjy1 = plane.data(2,j) + coord.ck.params(2,plane.data(9,j),plane.data(20,j))
pjx2 = plane.data(1,j) + coord.ck.params(3,plane.data(9,j),plane.data(20,j))
pjy2 = plane.data(2,j) + coord.ck.params(4,plane.data(9,j),plane.data(20,j))

'see if data blocks overlap; if so, adjust
IF ((pix1<=pjx1 AND pjx1<=pix2) OR (pix1<=pjx2 AND pjx2<=pix2)) AND ((piy1<=pjy1
AND pjy1<=piy2) OR (piy1<=pjy2 AND pjy2<=piy2)) GOTO ck.over.adjust
IF (pix1<=pjx AND pjx<=pix2) AND (piy1<=pjy AND pjy<=piy2) GOTO ck.over.adjust
GOTO ck.over.exit   :'no overlap, so no need to adjust

ck.over.adjust:   'plane i needs to adjust his data block position
orig.db.param = plane.data(9,i)   :'save original position
FOR kk = 1 TO 8   :'try it in each position, in preference order
```

```
  k = preferred.db.pos(kk,hdg.direction(1,plane.data(7,i)))
  plane.data(9,i)=k
  FOR l = 0 TO max.index   :'check for conflicts with all other planes
    IF l = i GOTO coa.exit   :'don't check the same plane stupid
    IF plane.data(0,l) <> 1 GOTO coa.exit   :'only active planes
    pix = plane.data(1,i)
    piy = plane.data(2,i)
    pjx = plane.data(1,l)
    pjy = plane.data(2,l)
    pix1 = plane.data(1,i) + coord.ck.params(1,plane.data(9,i),plane.data(20,i))
    piy1 = plane.data(2,i) + coord.ck.params(2,plane.data(9,i),plane.data(20,i))
    pix2 = plane.data(1,i) + coord.ck.params(3,plane.data(9,i),plane.data(20,i))
    piy2 = plane.data(2,i) + coord.ck.params(4,plane.data(9,i),plane.data(20,i))
    pjx1 = plane.data(1,l) + coord.ck.params(1,plane.data(9,l),plane.data(20,l))
    pjy1 = plane.data(2,l) + coord.ck.params(2,plane.data(9,l),plane.data(20,l))
    pjx2 = plane.data(1,l) + coord.ck.params(3,plane.data(9,l),plane.data(20,l))
    pjy2 = plane.data(2,l) + coord.ck.params(4,plane.data(9,l),plane.data(20,l))
    IF ((pix1<=pjx1 AND pjx1<=pix2) OR (pix1<=pjx2 AND pjx2<=pix2)) AND
((piy1<=pjy1 AND pjy1<=piy2) OR (piy1<=pjy2 AND pjy2<=piy2)) GOTO not.this.one
    IF (pix1<=pjx AND pjx<=pix2) AND (piy1<=pjy AND pjy<=piy2) GOTO not.this.one
    coa.exit:
  NEXT
  'this k was OK, so use it unless the data block would be off the screen
  IF (k=6 OR k=7 OR k=0) AND (plane.data(1,i) >= mac.screen.width-42) GOTO
not.this.one
  IF (k=4 OR k=3 OR k=2) AND (plane.data(1,i) <= 35) GOTO not.this.one
  IF (k=4 OR k=5 OR k=6) AND (plane.data(2,i) <= 25) GOTO not.this.one
  IF (k=2 OR k=1 OR k=0) AND (plane.data(2,i) >= mac.screen.height-52) GOTO
not.this.one
  GOTO use.this.k
  not.this.one:   'check the next K
NEXT
'no K was a good value, so exit
plane.data(9,i) = orig.db.param
GOTO ck.over.exit

use.this.k:   'update data block position on screen
thisk = k
plane.data(9,i) = orig.db.param
k = i: GOSUB paint.it.white
plane.data(9,i) = thisk
k = i: GOSUB paint.it.black  :'paint both overlapping planes
k = j: GOSUB paint.it.black
ck.over.exit:

RETURN

REM  ***********************************************
construct.response:   'build op response, but only allow n secs for input

mouseans$="": startt!=TIMER
WHILE TIMER-startt! < timelimit   :'allow n seconds to build the command
    a$=INKEY$   :'get 1 character from the keyboard
    'capture RETURN (13) and ENTER (3), meaning end of command
    IF a$ <> "" THEN
```

```
      IF ASC(a$)=13 OR ASC(a$)=3 THEN
        RETURN
      'capture delete key and erase one character from mouseans$ and screen
      ELSEIF ASC(a$) = 8 THEN
        IF LEN(mouseans$) > 0 THEN
          mouseans$ = LEFT$(mouseans$,LEN(mouseans$)-1)
          LOCATE 1,72+LEN(mouseans$): DrawText! " ": LOCATE 1,72+LEN(mouseans$)
        END IF
      'if space bar, ignore command and return
      ELSEIF ASC(a$) = 32 THEN
        mouseans$ = ""
        RETURN
      ELSE
      'if clear key pressed, code a c
        IF ASC(a$) = 27 THEN a$ = "c"
        DrawText! a$   :'echo key input on the screen
        mouseans$=mouseans$+a$    :'add to command string
      END IF
    END IF
WEND
'  time has expired, so beep and ignore any command partially entered
BEEP
mouseans$ = ""

RETURN

REM  ***********************************************
ctalk:   'speak the controllers commands to the planes

IF first.talk = 0 THEN tts$ =
carrier.names$(plane.data(3,mouseplane))+STR$(plane.data(4,mouseplane))+" , "
ELSE tts$=tts$+" and "
first.talk = 1
ON mcode GOTO ctalk.1, ctalk.2, ctalk.3, ctalk.4, ctalk.5, ctalk.6, ctalk.7,
ctalk.8, ctalk.9, ctalk.10

ctalk.1:
tts$=tts$+" you are cleared as requested "
GOTO ctalk.end

ctalk.2:
IF plane.data(5,mouseplane) = plane.data(10,mouseplane) THEN tts$=tts$+"
maintain "
IF plane.data(5,mouseplane) < plane.data(10,mouseplane) THEN tts$=tts$+" clime
to "
IF plane.data(5,mouseplane) > plane.data(10,mouseplane) THEN tts$=tts$+" de
scend to "
sayn! = plane.data(10,mouseplane)*100!
GOSUB saynum
tts$=tts$+sayn$+" feet"
GOTO ctalk.end

ctalk.3:
IF plane.data(6,mouseplane) = plane.data(11,mouseplane) THEN tts$=tts$+"
maintain speed of "
```

```
IF plane.data(6,mouseplane) < plane.data(11,mouseplane) THEN tts$=tts$+"
increese speed to "
IF plane.data(6,mouseplane) > plane.data(11,mouseplane) THEN tts$=tts$+" reduce
speed to "
sayn! = plane.data(11,mouseplane)*10
GOSUB saynum
tts$=tts$+sayn$+" knots"
GOTO ctalk.end

ctalk.4:
sayn! = plane.data(12,mouseplane)
GOSUB saynum
tts$=tts$+" turn to a heading of "+sayn$+" degrees"
GOTO ctalk.end

ctalk.5:
IF plane.data(17,mouseplane) <> 0 THEN tts$=tts$+" hold at your present poe
zishun" ELSE tts$=tts$+"you may proceed on course"
GOTO ctalk.end

ctalk.6:
tts$=tts$+" proceed to vore -- "+vor.names$(ABS(plane.data(12,mouseplane)))
GOTO ctalk.end

ctalk.7:
IF plane.data(5,mouseplane) = plane.data(10,mouseplane) THEN tts$=tts$+"
maintain "
IF plane.data(5,mouseplane) < plane.data(10,mouseplane) THEN tts$=tts$+" clime
to "
IF plane.data(5,mouseplane) > plane.data(10,mouseplane) THEN tts$=tts$+" de
scend to "
sayn! = plane.data(10,mouseplane)*100!
GOSUB saynum
tts$=tts$+sayn$+" feet, and "
IF plane.data(6,mouseplane) = plane.data(11,mouseplane) THEN tts$=tts$+"
maintain speed of "
IF plane.data(6,mouseplane) < plane.data(11,mouseplane) THEN tts$=tts$+"
increese speed to "
IF plane.data(6,mouseplane) > plane.data(11,mouseplane) THEN tts$=tts$+" reduce
speed to "
sayn! = plane.data(11,mouseplane)*10
GOSUB saynum
tts$=tts$+sayn$+" knots"
GOTO ctalk.end

ctalk.8:
tts$ = tts$ + " please divert to runway
"+jetway.check.name$(ABS(plane.data(15,mouseplane)))
tts$ = tts$ + " at " + airport$(ABS(plane.data(15,mouseplane))-8)
GOTO ctalk.end

ctalk.9:
IF plane.data(22,mouseplane) = 0 THEN tts$ = tts$ + " cancel approach clearance"
IF plane.data(22,mouseplane) = 1 THEN tts$ = tts$ + " you are cleared for the
approach to runway " + jetway.check.name$(ABS(plane.data(15,mouseplane)))
```

```
GOTO ctalk.end

ctalk.10:
IF plane.data(23,mouseplane) = 0 THEN tts$ = tts$ + " raydarr kontact": GOTO
ctalk.end
direc = plane.data(15,mouseplane)  :'requested heading
IF direc >= 0 THEN hdgdir = hdg.direction(1,direc) ELSE hdgdir = 0
currentx = plane.data(1,mouseplane)
p1=handoff.inset

IF hdgdir = 2 AND currentx>=mac.screen.width-4-p1 THEN wc = 2
IF hdgdir = 2 AND currentx<mac.screen.width-4-p1 THEN wc = 1
IF hdgdir = 4 AND currentx>=mac.screen.width-4-p1 THEN wc = 2
IF hdgdir = 4 AND currentx<mac.screen.width-4-p1 THEN wc = 3
IF hdgdir = 6 AND currentx<=p1 THEN wc = 4
IF hdgdir = 6 AND currentx>p1 THEN wc = 3
IF hdgdir = 8 AND currentx<=p1 THEN wc = 4
IF hdgdir = 8 AND currentx>p1 THEN wc = 1

IF hdgdir = 1 THEN wc = 1
IF hdgdir = 3 THEN wc = 2
IF hdgdir = 5 THEN wc = 3
IF hdgdir = 7 THEN wc = 4
IF direc = -1 THEN wc = 1
IF direc = -2 THEN wc = 3
IF direc = -3 THEN wc = 2
IF direc = -4 THEN wc = 4
IF direc = -5 AND aw4ep$ = "N" THEN wc = 1
IF direc = -5 AND aw4ep$ = "E" THEN wc = 2
IF direc = -6 THEN wc = 4
IF direc = -7 AND aw13ep$ = "S" THEN wc = 3
IF direc = -7 AND aw13ep$ = "E" THEN wc = 2
IF direc = -8 THEN wc = 1

IF plane.data(23,mouseplane) = 1 THEN tts$ = tts$ + " kontact " + cen$(wc) + "
center on " + freq$(wc)
GOTO ctalk.end

ctalk.end:
RETURN

REM  ***********************************************
data.block.adjust:   'check for overlapping data blocks and adjust

IF debug.switch = -1 THEN RETURN

IF params(3) = 1 THEN RETURN   :'don't adjust in failure mode
FOR i = 0 TO max.index
  IF plane.data(0,i) <> 1 GOTO dba.exit1  :'only for active planes
  FOR j = 0 TO max.index
    IF j = i GOTO dba.exit2
    IF plane.data(0,j) <> 1 GOTO dba.exit2
    GOSUB check.overlap
    dba.exit2:
  NEXT
  dba.exit1:
NEXT

RETURN

REM  ***********************************************
delayNsecs:    'delay for N seconds, allowing mouse clicks

garb = FRE(0)    :'string compression, do it while waiting
IF params(13) < 1 THEN RETURN
TIMER ON
ON TIMER (params(13)) GOSUB stop.Nsec.delay
delay.switch = 0
MOUSE ON: MENU ON   :'allow these while waiting
WHILE delay.switch = 0:  WEND
MOUSE STOP: MENU STOP   :'suspend these now
RETURN

REM  ***********************************************
demo.proc:   'issue commands automatically in demo mode

'  this section will issue controller commands automatically
'  if the simulation is being operated in demonstration mode.
'  it won't handle each and every situation perfectly, but it
'  will attempt to handle all aircraft in the following manner:
'
'  (1) issue altitude command if assigned alt <> requested alt
'  (2) issue speed command if assigned spd <> requested spd
'  (3) issue heading command if assigned hdg <> requested hdg and
'      requested hdg is degrees
'  (4) issue proper heading command if requested hdg = jetway and
'      assigned hdg <> final hdg and assigned hdg <> right transit hdg
'  (5) clear for approach if requesting runway and not already cleared
'  (6) accept a handoff for all new planes
'  (7) initiate a handoff at the proper time for outbound planes

FOR dp = 0 TO max.index
  ctrlr.cmd$=""   :'this holds the controller commands
  tts$="": first.talk = 0   :'set to 1 if command will be given
  FOR tx = 0 TO plane.params: old.plane.data(tx)=plane.data(tx,dp):NEXT

  IF plane.data(0,dp) <> 1 GOTO demo.exit   :'look at active ones only

  '  case 6 - accept handoff

  IF plane.data(23,dp) <> 0 GOTO demo0
  mouseplane=dp: mcode=10: GOSUB ctalk: plane.data(23,dp)=1
  ctrlr.cmd$=ctrlr.cmd$+"/0"
  demo0:

  '  case 7 - initiate handoff

  IF plane.data(15,dp) < -8 GOTO demoz2
  IF plane.data(23,dp) = 2 GOTO demoz2
  direc = plane.data(15,dp)  :'requested heading
  currx = plane.data(1,dp): curry = plane.data(2,dp)
  p1=handoff.inset
  IF direc >= 0 THEN hdgdir = hdg.direction(1,direc) ELSE hdgdir = 0

  IF hdgdir = 1 AND handoff.status(1) = 1 GOTO demoz2
  IF hdgdir = 5 AND handoff.status(2) = 1 GOTO demoz2
  IF hdgdir = 3 AND handoff.status(3) = 1 GOTO demoz2
  IF hdgdir = 7 AND handoff.status(4) = 1 GOTO demoz2
  IF hdgdir = 2 AND (handoff.status(1) = 1 AND handoff.status(3) = 1) GOTO
demoz2
  IF hdgdir = 4 AND (handoff.status(3) = 1 AND handoff.status(2) = 1) GOTO
demoz2
  IF hdgdir = 6 AND (handoff.status(2) = 1 AND handoff.status(4) = 1) GOTO
demoz2
  IF hdgdir = 8 AND (handoff.status(1) = 1 AND handoff.status(4) = 1) GOTO
demoz2
  IF direc = -1 AND handoff.status(1) = 1 GOTO demoz2
  IF direc = -2 AND handoff.status(2) = 1 GOTO demoz2
  IF direc = -3 AND handoff.status(3) = 1 GOTO demoz2
  IF direc = -4 AND handoff.status(4) = 1 GOTO demoz2
  IF direc = -5 AND aw4ep$ = "N" AND handoff.status(1) = 1 GOTO demoz2
  IF direc = -5 AND aw4ep$ = "E" AND handoff.status(3) = 1 GOTO demoz2
  IF direc = -6 AND handoff.status(4) = 1 GOTO demoz2
  IF direc = -7 AND aw13ep$ = "S" AND handoff.status(2) = 1 GOTO demoz2
  IF direc = -7 AND aw13ep$ = "E" AND handoff.status(3) = 1 GOTO demoz2
  IF direc = -8 AND handoff.status(1) = 1 GOTO demoz2

  IF hdgdir = 1 AND (curry<=p1) GOTO demoz1
  IF hdgdir = 2 AND ((curry<=p1) OR (currx>=mac.screen.width-4-p1)) GOTO demoz1
  IF hdgdir = 3 AND (currx>=mac.screen.width-4-p1) GOTO demoz1
  IF hdgdir = 4 AND ((currx>=mac.screen.width-4-p1) OR
(curry>=mac.screen.height-26-p1)) GOTO demoz1
  IF hdgdir = 5 AND (curry>=mac.screen.height-26-p1) GOTO demoz1
  IF hdgdir = 6 AND ((curry>=mac.screen.height-26-p1) OR (currx<=p1)) GOTO
demoz1
  IF hdgdir = 7 AND (currx<=p1) GOTO demoz1
  IF hdgdir = 8 AND ((curry<=p1) OR (currx<=p1)) GOTO demoz1

  IF direc = -1 AND (curry<=p1) GOTO demoz1
  IF direc = -2 AND (curry>=mac.screen.height-26-p1) GOTO demoz1
  IF direc = -3 AND (currx>=mac.screen.width-4-p1) GOTO demoz1
  IF direc = -4 AND (currx<=p1) GOTO demoz1
  IF direc = -5 AND aw4ep$ = "N" AND (curry<=p1) GOTO demoz1
  IF direc = -5 AND aw4ep$ = "E" AND (currx>=mac.screen.width-4-p1) GOTO demoz1
  IF direc = -6 AND (currx<=p1) GOTO demoz1
  IF direc = -7 AND aw13ep$ = "S" AND (curry>=mac.screen.height-26-p1) GOTO
demoz1
  IF direc = -7 AND aw13ep$ = "E" AND (currx>=mac.screen.width-4-p1) GOTO demoz1
  IF direc = -8 AND (curry<=p1) GOTO demoz1
  GOTO demoz2

  demoz1:
  mouseplane=dp: mcode=10: GOSUB ctalk: plane.data(23,dp)=2
  ctrlr.cmd$=ctrlr.cmd$+"/0"
  demoz2:

  '  case 1 - altitude change

  IF plane.data(10,dp) = plane.data(13,dp) GOTO demo1
  IF plane.data(15,dp) < -8 GOTO demo1
  mouseplane=dp: mcode=2: plane.data(10,dp)=plane.data(13,dp)
  GOSUB ctalk

ctrlr.cmd$=ctrlr.cmd$+"/A"+RIGHT$(STR$(plane.data(10,dp)),LEN(STR$(plane.data(10
,dp)))-1)
  demo1:

  '  case 2 - speed change

  IF plane.data(11,dp) = plane.data(14,dp) GOTO demo2
  IF plane.data(15,dp) < -8 GOTO demo2
  mouseplane=dp: mcode=3: plane.data(11,dp)=plane.data(14,dp)
  GOSUB ctalk

ctrlr.cmd$=ctrlr.cmd$+"/S"+RIGHT$(STR$(plane.data(11,dp)),LEN(STR$(plane.data(11
,dp)))-1)
  demo2:

  '  case 3 - degree heading change

  IF plane.data(15,dp) < 0 GOTO demo3
  IF plane.data(12,dp) = plane.data(15,dp) GOTO demo3
  mouseplane=dp: mcode=4: plane.data(12,dp)=plane.data(15,dp)
  GOSUB ctalk

ctrlr.cmd$=ctrlr.cmd$+"/H"+RIGHT$(STR$(plane.data(12,dp)),LEN(STR$(plane.data(12
,dp)))-1)
  demo3:

'   case 4 - jetway intercept

  IF plane.data(15,dp) >= 0 GOTO demo4
  IF plane.data(15,dp) < -4 GOTO demo4  :'ignore runways and irregular jetways
  IF plane.data(12,dp) = jetway.check.data(ABS(plane.data(15,dp))) GOTO demo4
:'on final heading
  IF plane.data(15,dp) = -1 AND plane.data(12,dp) = 90 AND plane.data(1,dp) <
aw3618x1 GOTO demo4
  IF plane.data(15,dp) = -1 AND plane.data(12,dp) = 270 AND plane.data(1,dp) >
aw3618x1 GOTO demo4
  IF plane.data(15,dp) = -2 AND plane.data(12,dp) = 90 AND plane.data(1,dp) <
aw3618x1 GOTO demo4
  IF plane.data(15,dp) = -2 AND plane.data(12,dp) = 270 AND plane.data(1,dp) >
aw3618x1 GOTO demo4
  IF plane.data(15,dp) = -3 AND plane.data(12,dp) = 0 AND plane.data(2,dp) >
aw0927y1 GOTO demo4
  IF plane.data(15,dp) = -3 AND plane.data(12,dp) = 180 AND plane.data(2,dp) <
aw0927y1 GOTO demo4
  IF plane.data(15,dp) = -4 AND plane.data(12,dp) = 0 AND plane.data(2,dp) >
aw0927y1 GOTO demo4
```

```
  IF plane.data(15,dp) = -4 AND plane.data(12,dp) = 180 AND plane.data(2,dp) <
aw0927y1 GOTO demo4

  IF (plane.data(15,dp) = -1 OR plane.data(15,dp) = -2) AND plane.data(1,dp) <
aw3618x1 THEN transit.hdg = 90
  IF (plane.data(15,dp) = -1 OR plane.data(15,dp) = -2) AND plane.data(1,dp) >
aw3618x1 THEN transit.hdg = 270
  IF (plane.data(15,dp) = -3 OR plane.data(15,dp) = -4) AND plane.data(2,dp) <
aw0927y1 THEN transit.hdg = 180
  IF (plane.data(15,dp) = -3 OR plane.data(15,dp) = -4) AND plane.data(2,dp) >
aw0927y1 THEN transit.hdg = 0

  mouseplane=dp: mcode=4: plane.data(12,dp)=transit.hdg
  GOSUB ctalk

ctrlr.cmd$=ctrlr.cmd$+"/H"+RIGHT$(STR$(plane.data(12,dp)),LEN(STR$(plane.data(12
,dp)))-1)
  demo4:

  '   case 5 - clear runway bound planes for the approach

  IF plane.data(15,dp) > -9 GOTO demo5
  IF plane.data(22,dp) = 1 GOTO demo5    :'already cleared for approach
  mouseplane=dp: mcode=9:
plane.data(12,dp)=plane.data(15,dp)+8:plane.data(22,dp)=1
  GOSUB ctalk
  ctrlr.cmd$=ctrlr.cmd$+"/C"
  demo5:

  IF first.talk = 0 GOTO demo.exit
  mousez = dp: GOSUB build.cmd.prompt
  LOCATE 1,1
  DrawText! prompt$
  LOCATE 1,68
  DrawText! "CMD:"+MID$(ctrlr.cmd$,2,15)

  voice.type = -1: GOSUB speakit

  WINDOW 1
  mouseplane = dp: GOSUB redraw.one.plane
  demo.exit:

NEXT

RETURN

REM  **********************************************
dialogproc:   'handle dialog entries in the options screen

dialog.num = DIALOG(0)
IF dialog.num = 1 THEN GOSUB handle.button: GOTO end.dialog
IF dialog.num = 2 THEN GOSUB handle.editfield: GOTO end.dialog

end.dialog:
RETURN
```

```
REM **********************************************
display.command.help:   'display the command help screen

WINDOW 3,,(2,22)-(mac.screen.width-2,mac.screen.height-4),3
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
lcnt = 3: LOCATE lcnt,5
FOR dch = 1 TO 24
  DrawText! command.help$(dch)
  lcnt = lcnt + 1: LOCATE lcnt,5
NEXT
MENU ON: MOUSE ON
xwait = 1: WHILE xwait = 1: WEND
RETURN

REM **********************************************
display.options:   'display the options screen

WINDOW 3,,(2,22)-(510,338),-3   :'modal box, can't click outside on menus
GOSUB build.options.screen   :'establish all buttons and edit fields

DIALOG ON  :'enable dialogs
xwait = 1: WHILE xwait = 1: WEND   :'continue button will let you continue
WINDOW CLOSE 3
IF options = 0 THEN WINDOW 1: GOSUB paint.with.clear
DIALOG OFF: MOUSE ON: MENU ON   :'disable dialogs, reenable mouse/menus
RETURN

REM **********************************************
display.ref.screen:   'display the reference screen, with all features labelled

WINDOW 3,,(2,22)-(mac.screen.width-2,mac.screen.height-4),3
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
GOSUB draw.airways.ref
FOR drs = 1 TO 15
  LOCATE ref.screen(1,drs), ref.screen(2,drs): DrawText! ref.screen$(drs)
NEXT
MENU ON: MOUSE ON
xwait = 1: WHILE xwait = 1: WEND
RETURN

REM **********************************************
display.score:   'display the score screen

GOSUB figure.score
WINDOW 3,,(2,22)-(mac.screen.width-2,mac.screen.height-4),3
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
lcnt = 1: LOCATE lcnt,15
DrawText! "AIR TRAFFIC CONTROL -- OPERATIONAL STATISTICS"
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "DATE:  " + start.date$
lcnt = lcnt + 1: LOCATE lcnt,10
```

```
DrawText! "START TIME:  " + start.time$
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "STOP TIME:  " + stop.time$
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "MAXIMUM PLANES HANDLED AT ONE TIME  =  " + STR$(max.planes.handled)
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "TOTAL PLANES EXITED SUCCESSFULLY  =  " +
STR$(planes.exited.successfully)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL PLANES LANDED SUCCESSFULLY  =  " +
STR$(planes.landed.successfully)
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "TOTAL PLANES EXITED AT IMPROPER ALTITUDE  =  " +
STR$(planes.exited.bad.alt)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL PLANES EXITED AT IMPROPER SPEED  =  " +
STR$(planes.exited.bad.spd)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL PLANES EXITED AT IMPROPER HEADING  =  " +
STR$(planes.exited.bad.hdg)
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "TOTAL PLANES EXITED WITHOUT PROPER HANDOFF  =  " +
STR$(planes.exited.bad.sector)
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "TOTAL PLANES CRASHED  =  " + STR$(planes.crashed.total)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL PROXIMITY WARNINGS GIVEN  =  " +
STR$(planes.proximity.warnings)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL CONTROL ZONE VIOLATIONS  =  " + STR$(planes.tca.violations)
lcnt = lcnt + 1: LOCATE lcnt,10
DrawText! "TOTAL COMPUTER FAILURE CYCLES  =  " + STR$(cf.cycles)
lcnt = lcnt + 2: LOCATE lcnt,10
DrawText! "TOTAL SCORE = " + STR$(ts) + " POINTS"
lcnt = lcnt + 2: LOCATE lcnt,10
IF ts < 25 THEN DrawText! "YOU MAKE ME WANNA PUKE, YOU KNOW IT?"
IF ts >= 25 AND ts <= 49 THEN DrawText! "YOU'RE A MENACE TO THE FLYING PUBLIC."
IF ts >= 50 AND ts <= 74 THEN DrawText! "YOU COULD STAND SOME IMPROVEMENT, YOU
KNOW?"
IF ts >= 75 AND ts <= 99 THEN DrawText! "KEEP TRYING, YOU'RE GETTING THE HANG OF
IT."
IF ts >= 100 AND ts <= 124 THEN DrawText! "THAT'S NOT TOO BAD; YOU'RE CATCHING
ON NOW."
IF ts >= 125 AND ts <= 149 THEN DrawText! "PRETTY GOOD JOB; I'M IMPRESSED!"
IF ts >= 150 AND ts <= 174 THEN DrawText! "YOU SHOW REAL TALENT, YOU KNOW IT?"
IF ts >= 175 AND ts <= 199 THEN DrawText! "YOU SHOULD DO THIS PROFESSIONALLY,
YOU KNOW IT?"
IF ts >= 200 THEN DrawText! "YOU HAVE OBVIOUSLY BEEN DOING THIS AWHILE, HAVEN'T
YOU?"
MENU ON: MOUSE ON
xwait = 1: WHILE xwait = 1: WEND
RETURN

REM  ***********************************************
display.status:   'display the status screen
```

```
WINDOW 3,,(2,22)-(mac.screen.width-2,mac.screen.height-4),3
GOSUB figure.score
CALL TEXTFONT (4)    :'monaco font
CALL TEXTSIZE (9)    :'small type
LINE (250,0)-(250,mac.screen.height-26)
LOCATE 1,2: DrawText! "PLANE  ASG REQ   FLIGHT STATUS"
LOCATE 2,2: DrawText! "IDENT  HDG HDG"
lcnt = 2
large.planes=0: small.planes=0: unident.planes=0
FOR g = 1 TO 13: req.hdgs(g)=0: NEXT
FOR disp.idx = 0 TO max.index
  IF plane.data(0,disp.idx) = 0 GOTO stat1  :'inactive
  IF plane.data(0,disp.idx) = 2 THEN unident.planes = unident.planes + 1: GOTO
stat1
  IF carrier.size(plane.data(3,disp.idx)) = 0 THEN small.planes = small.planes +
1 ELSE large.planes = large.planes + 1
  IF plane.data(15,disp.idx) > 0 THEN req.hdgs(13) = req.hdgs(13) + 1 ELSE
req.hdgs(ABS(plane.data(15,disp.idx))) = req.hdgs(ABS(plane.data(15,disp.idx)))
+ 1
  IF params(3) = 1 GOTO stat1
  lcnt = lcnt+1: LOCATE lcnt,2
  DrawText! carrier$(plane.data(3,disp.idx))+" "
  DrawText! RIGHT$("00"+MID$(STR$(plane.data(4,disp.idx)),2,3),3)
  dhdg2$ = RIGHT$("00"+MID$(STR$(plane.data(12,disp.idx)),2,3),3)
  dhdg3$ = RIGHT$("00"+MID$(STR$(plane.data(15,disp.idx)),2,3),3)
  IF plane.data(12,disp.idx)  <  0 THEN dhdg2$ =
vor.names$(ABS(plane.data(12,disp.idx)))
  IF plane.data(15,disp.idx)  <  0 THEN dhdg3$ =
jetway.check.name$(ABS(plane.data(15,disp.idx)))
  DrawText! " "+dhdg2$+" "+dhdg3$+" - "
  IF plane.data(17,disp.idx) = -1 THEN DrawText! "HOLDING INDEFINITELY": GOTO
stat1
  IF plane.data(17,disp.idx) > 0 THEN DrawText! "HOLDING
FOR"+STR$(plane.data(17,disp.idx))+" CYCLES": GOTO stat1
  IF plane.data(15,disp.idx) >= -8 GOTO stat2
  IF plane.data(22,disp.idx) = 1 THEN smsg$ = "CLEARED FOR APPROACH" ELSE smsg$
= "RUNWAY BOUND"
  IF plane.data(12,disp.idx) <> jetway.check.data(ABS(plane.data(15,disp.idx)))
THEN DrawText! smsg$: GOTO stat1
  IF plane.data(1,disp.idx) < rwarea(1,ABS(plane.data(15,disp.idx))-8) OR
plane.data(1,disp.idx) > rwarea(2,ABS(plane.data(15,disp.idx))-8) THEN DrawText!
smsg$: GOTO stat1
  IF plane.data(2,disp.idx) < rwarea(3,ABS(plane.data(15,disp.idx))-8) OR
plane.data(2,disp.idx) > rwarea(4,ABS(plane.data(15,disp.idx))-8) THEN DrawText!
smsg$: GOTO stat1
  DrawText! "ON FINAL APPROACH": GOTO stat1
  stat2:
  IF plane.data(10,disp.idx) <> plane.data(13,disp.idx) THEN DrawText! "ALTITUDE
ERROR": GOTO stat1
  IF plane.data(11,disp.idx) <> plane.data(14,disp.idx) THEN DrawText! "SPEED
ERROR": GOTO stat1
  IF plane.data(15,disp.idx) >= 0 AND plane.data(12,disp.idx) <>
plane.data(15,disp.idx) THEN DrawText! "HEADING ERROR": GOTO stat1
  IF plane.data(15,disp.idx) >= 0 THEN DrawText! "OK": GOTO stat1
  IF plane.data(12,disp.idx) <> jetway.check.data(ABS(plane.data(15,disp.idx)))
THEN DrawText! "ENROUTE TO AIRWAY": GOTO stat1
  IF (plane.data(15,disp.idx) = -1 OR plane.data(15,disp.idx) = -2) AND
plane.data(1,disp.idx) >= aw3618x1-10 AND plane.data(1,disp.idx) <= aw3618x1+10
THEN DrawText! "ON AIRWAY": GOTO stat1
  IF (plane.data(15,disp.idx) = -3 OR plane.data(15,disp.idx) = -4) AND
plane.data(2,disp.idx) >= aw0927y1-10 AND plane.data(2,disp.idx) <= aw0927y1+10
THEN DrawText! "ON AIRWAY": GOTO stat1
  sumxy = plane.data(1,disp.idx) + plane.data(2,disp.idx)
  difxy = plane.data(1,disp.idx) - plane.data(2,disp.idx)
  IF (plane.data(15,disp.idx) = -5 OR plane.data(15,disp.idx) = -6) AND sumxy >=
aw0422y1-10 AND sumxy <= aw0422y1+10 THEN DrawText! "ON AIRWAY": GOTO stat1
  IF (plane.data(15,disp.idx) = -7 OR plane.data(15,disp.idx) = -8) AND difxy >=
aw1331x1-10 AND difxy <= aw1331x1+10 THEN DrawText! "ON AIRWAY": GOTO stat1
  DrawText! "ENROUTE TO AIRWAY"

  stat1:
NEXT

LINE (0,21)-(250,21)
LINE (395,0)-(395,180)
LOCATE 1,44: DrawText! "MINUTES PLAYED: "+STR$(nummins)
LOCATE 2,44: DrawText! "PLANES EXITED:
"+STR$(planes.exited.successfully+peu+planes.exited.bad.sector)
LOCATE 3,44: DrawText! "PLANES LANDED: "+STR$(planes.landed.successfully)
LOCATE 4,44: DrawText! "SCORE: "+STR$(ts)
LINE (250,47)-(395,47)
LOCATE 6,44: DrawText! "POSITION: "+mode.msg$(params(6))
LOCATE 7,44: DrawText! "SKILL LEVEL: "+STR$(params(5))
LOCATE 8,44: DrawText! "AUTO-TURN: ": IF params(2) = 1 THEN DrawText! "ON" ELSE
DrawText! "OFF"
LOCATE 9,44: DrawText! "FAILURE MODE: ": IF params(3) = 0 THEN DrawText! "OFF"
ELSE DrawText! "ON"
LOCATE 10,44: DrawText! "VOICE: "
IF params(4) = 1 THEN DrawText! "ON" ELSE DrawText! "OFF"
LOCATE 11,44: DrawText! "CENTER: "+center.name$
LINE (250,125)-(395,125)
LOCATE 13,44: DrawText! "LARGE PLANES: "+STR$(large.planes)
LOCATE 14,44: DrawText! "SMALL PLANES: "+STR$(small.planes)
LOCATE 15,44: DrawText! "UNIDENT PLANES: "+STR$(unident.planes)
LOCATE 16,44: DrawText! "TOTAL PLANES:
"+STR$(large.planes+small.planes+unident.planes)
LINE (250,180)-(mac.screen.width-4,180)
LOCATE 18,44: DrawText! "REQUESTED HEADINGS:"
FOR disp.idx = 1 TO 13
  LOCATE stat.screen(1,disp.idx), stat.screen(2,disp.idx)
  DrawText! stat.screen$(disp.idx)+STR$(req.hdgs(stat.screen(3,disp.idx)))
NEXT
LINE (250,268)-(mac.screen.width-4,268)
LOCATE 26,44
DrawText! "EXITS: NORTH-"
IF handoff.status(1) = 0 THEN DrawText! "OK" ELSE DrawText! "NO"
DrawText! " SOUTH-"
IF handoff.status(2) = 0 THEN DrawText! "OK" ELSE DrawText! "NO"
DrawText! " EAST-"
```

```
IF handoff.status(3) = 0 THEN DrawText! "OK" ELSE DrawText! "NO"
DrawText! " WEST-"
IF handoff.status(4) = 0 THEN DrawText! "OK" ELSE DrawText! "NO"
LOCATE 27,44
DrawText! airport$(1)+": 36-"
IF airport.status(1) = 0 THEN DrawText! "OPEN" ELSE DrawText! "CLOSED"
DrawText! " 18-"
IF airport.status(2) = 0 THEN DrawText! "OPEN" ELSE DrawText! "CLOSED"
LOCATE 28,44
DrawText! airport$(3)+": 27-"
IF airport.status(3) = 0 THEN DrawText! "OPEN" ELSE DrawText! "CLOSED"
DrawText! " 09-"
IF airport.status(4) = 0 THEN DrawText! "OPEN" ELSE DrawText! "CLOSED"
FOR disp.idx = 1 TO 16
  LOCATE stat.screen1(1,disp.idx), stat.screen1(2,disp.idx)
  DrawText! stat.screen1$(disp.idx)
NEXT
MENU ON: MOUSE ON
xwait = 1: WHILE xwait = 1: WEND
RETURN

REM  ***********************************************
dothechecks:   'perform these checks during every update cycle
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB data.block.adjust     :'don't let data blocks overlap
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB at.the.jetway.check  :'print msg if plane is at desired jetway
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB landing.check        :'print msg if planes lands on runway
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB set.cfa.hdg            :'if cleared for approach, handle arrival at VOR
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB set.vor.hdg            :'set heading if VOR-bound
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB proximity.check  :'check for planes too close at the same altitude & warn
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB advisory.check   :'check for long-term conflicts
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB tca.violation.check  :'check for control zone violaters
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB release.from.hold    :'check to see if it is time to stop holding
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB handle.handoff   :'close airports or handoffs
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB adjust.cfa.alt.speed    :'adjust alt and speed if cleared for approach
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
GOSUB specl.requests   :'generate special requests from planes or add new planes
MOUSE ON: MENU ON: MOUSE STOP: MENU STOP
RETURN

REM  ***********************************************
drag.db:    :'reposition the data block by dragging it

'    variables mouseex and mouseey contain end-drag coordinates
'    see if they fall at a valid position to move the data block to
'    if so, then move the data block there; if not, beep once

FOR t1 = 0 TO 7    :'data block offset direction
  FOR t2 = 0 TO 15    :'leader length factor

    xxx1 = plane.data(1,mousez) + coord.ck.params(1,t1,t2)
    yyy1 = plane.data(2,mousez) + coord.ck.params(2,t1,t2)
    xxx2 = plane.data(1,mousez) + coord.ck.params(3,t1,t2)
    yyy2 = plane.data(2,mousez) + coord.ck.params(4,t1,t2)
    IF (mouseex >= xxx1 AND mouseex <= xxx2) AND (mouseey >= yyy1 AND mouseey <=
yyy2) THEN GOSUB move.db: GOTO drag.db.exit

  NEXT
NEXT
BEEP    :'dragged to an impossible position

drag.db.exit:

RETURN

REM  ***********************************************
draw.airways:   'print airways, runways, VORs, and control zones on the screen
gr.color = 33   :'black

IF dswt.a = 0 GOTO skip.airway.draw
LINE (aw0422x1,aw0422y1)-(aw0422x2,aw0422y2),gr.color  :'J22 and J4
LINE (aw0927x1,aw0927y1)-(aw0927x2,aw0927y2),gr.color  :'J9 and J27
LINE (aw3618x1,aw3618y1)-(aw3618x2,aw3618y2),gr.color  :'J36 and J18
LINE (aw1331x1,aw1331y1)-(aw1331x2,aw1331y2) ,gr.color :'J31 and J13
skip.airway.draw:
IF dswt.r = 0 GOTO skip.runway.draw
LINE (ap2x,ap2y)-(ap2x+30,ap2y+2),gr.color,BF   :'R27 and R09
LINE (ap1x,ap1y)-(ap1x+2,ap1y+30),gr.color,BF    :'R36 and R18
skip.runway.draw:
IF dswt.rb = 0 GOTO skip.rbarricade.draw
IF airport.status(1) = 1 THEN LINE (ap1x-2,ap1y+33)-(ap1x+4,ap1y+35),gr.color,BF
IF airport.status(2) = 1 THEN LINE (ap1x-2,ap1y-5)-(ap1x+4,ap1y-3),gr.color,BF
IF airport.status(3) = 1 THEN LINE (ap2x+33,ap2y-2)-(ap2x+35,ap2y+4),gr.color,BF
IF airport.status(4) = 1 THEN LINE (ap2x-5,ap2y-2)-(ap2x-3,ap2y+4),gr.color,BF
skip.rbarricade.draw:
IF dswt.v = 0 GOTO skip.vor.draw
CIRCLE (ap2x-40,ap2y+1),3,gr.color    :'VOR V09
CIRCLE (ap2x+70,ap2y+1),3,gr.color    :'VOR V27
CIRCLE (ap1x+1,ap1y-40),3,gr.color    :'VOR V18
CIRCLE (ap1x+1,ap1y+70),3,gr.color    :'VOR V36
skip.vor.draw:
IF dswt.cz = 0 GOTO skip.cz.draw
LINE (ap2x-60,ap2y-39)-(ap2x+90,ap2y+41),gr.color,B    :'control zone 1
LINE (ap1x-39,ap1y-60)-(ap1x+41,ap1y+90),gr.color,B    :'control zone 2
skip.cz.draw:
IF dswt.hb = 0 GOTO skip.hbarricade.draw
IF handoff.status(1) = 1 THEN LINE (0,0)-(mac.screen.width-4,7),gr.color,BF
IF handoff.status(2) = 1 THEN LINE (0,mac.screen.height-35)-(mac.screen.width-
4,mac.screen.height-26),gr.color,BF
```

```
IF handoff.status(3) = 1 THEN LINE (mac.screen.width-12,0)-(mac.screen.width-
4,mac.screen.height-26),gr.color,BF
IF handoff.status(4) = 1 THEN LINE (0,0)-(7,mac.screen.height-26),gr.color,BF
skip.hbarricade.draw:
IF dswt.hz <> 0 THEN GOSUB draw.handoff.zone
IF dswt.g <> 0 THEN GOSUB draw.geography

RETURN

REM  ***********************************************
draw.airways.ref:   'print airways for reference screen only
gr.color = 33   :'black
LINE (0,237)-(237,0),gr.color  :'J22 and J4
LINE (0,150)-(508,150),gr.color  :'J9 and J27
LINE (370,0)-(370,316),gr.color  :'J36 and J18
LINE (100,0)-(416,316) ,gr.color :'J31 and J13
LINE (150,240)-(180,242),gr.color,BF   :'R27 and R09
LINE (420,75)-(422,105),gr.color,BF    :'R36 and R18
CIRCLE (110,241),3,gr.color    :'VOR V09
CIRCLE (220,241),3,gr.color    :'VOR V27
CIRCLE (421,35),3,gr.color    :'VOR V18
CIRCLE (421,145),3,gr.color    :'VOR V36
LINE (90,201)-(240,281),gr.color,B    :'control zone 1
LINE (381,15)-(461,165),gr.color,B    :'control zone 2
GOSUB draw.handoff.zone

RETURN

REM  ***********************************************
draw.geography:   'draw user-defined geography features

IF num.glines <= 0 GOTO draw.geo.text
FOR dg = 1 TO num.glines
  FOR dgg = 1 TO num.pts(dg)-1
    LINE (geo.points(1,dgg,dg),geo.points(2,dgg,dg))-
(geo.points(1,dgg+1,dg),geo.points(2,dgg+1,dg)),gr.color
  NEXT
NEXT

draw.geo.text:
IF num.geo.text <=0 THEN RETURN
CALL TEXTMODE (1)
FOR dg = 1 TO num.geo.text
  CALL MOVETO (geo.text.loc(1,dg),geo.text.loc(2,dg))
  DrawText! geo.text$(dg)
NEXT

RETURN

REM  ***********************************************
draw.handoff.zone:   'draw the dotted lines for the handoff zone

lx=0:ly=0:ux=mac.screen.width-4:uy=mac.screen.height-26

p1=handoff.inset :p2=handoff.len :p3=handoff.dashsx :p4=handoff.dashsy
```

```
LINE (lx+p1,ly+p1)-(lx+p1+p2,ly+p1),gr.color
LINE (lx+p1,uy-p1)-(lx+p1+p2,uy-p1),gr.color
LINE (ux-p1,ly+p1)-(ux-p1-p2,ly+p1),gr.color
LINE (ux-p1,uy-p1)-(ux-p1-p2,uy-p1),gr.color

LINE (lx+p1,ly+p1)-(lx+p1,ly+p1+p2),gr.color
LINE (lx+p1,uy-p1)-(lx+p1,uy-p1-p2),gr.color
LINE (ux-p1,ly+p1)-(ux-p1,ly+p1+p2),gr.color
LINE (ux-p1,uy-p1)-(ux-p1,uy-p1-p2),gr.color

xws=(((ux-p1-p2)-(lx+p1+p2))-(p3*p2))/(p3+1)
yws=(((uy-p1-p2)-(ly+p1+p2))-(p4*p2))/(p4+1)

hox=lx+p1+p2
FOR hoi = 1 TO p3
  LINE(hox+xws,ly+p1)-(hox+xws+p2,ly+p1),gr.color
  LINE(hox+xws,uy-p1)-(hox+xws+p2,uy-p1),gr.color
  hox=hox+xws+p2
NEXT

hoy=ly+p1+p2
FOR hoi = 1 TO p4
  LINE(lx+p1,hoy+yws)-(lx+p1,hoy+yws+p2),gr.color
  LINE(ux-p1,hoy+yws)-(ux-p1,hoy+yws+p2),gr.color
  hoy=hoy+yws+p2
NEXT

RETURN

REM  ***********************************************
figure.score:   'calculate the current score, put in ts

stop.time$ = TIME$
ts = planes.exited.successfully + max.planes.handled
ts = ts + (2*planes.landed.successfully)
ts = ts - (20*planes.crashed.total) - planes.proximity.warnings -
planes.tca.violations
peu = planes.exited.bad.alt + planes.exited.bad.spd + planes.exited.bad.hdg
ts = ts - (2*peu)
ts = ts - (5*planes.exited.bad.sector)
ts = ts + INT(cf.cycles/5)
IF peu = 0 THEN ts = ts + 5
min1 = VAL(MID$(stop.time$,4,2))
min2 = VAL(MID$(start.time$,4,2))
hour1 = VAL(LEFT$(stop.time$,2))
hour2 = VAL(LEFT$(start.time$,2))
IF hour1 < hour2 THEN hour1 = hour1 + 24
IF min1 < min2 THEN min1 = min1 + 60: hour1 = hour1 - 1
nummins = min1 - min2 + (60*(hour1 - hour2))
ts = ts + nummins
RETURN

REM  ***********************************************
gen.comp.failure:   'generate a computer failure
```

```
IF params(3) = 0 THEN
  params(3) = 1
  IF params(4) = 1 THEN BEEP: BEEP: BEEP: BEEP: BEEP
  tts$ = "computer failure computer failure"
  voice.type = -1
  GOSUB speakit
  GOSUB paint.with.clear
ELSE
  params(3) = 0
  tts$ = "computer enaybled"
  voice.type = -1
  GOSUB speakit
  GOSUB paint.with.clear
END IF
RETURN

REM  ***********************************************
get.response:   'get a command from the operator

'save original values
FOR tx = 0 TO plane.params: old.plane.data(tx)=plane.data(tx,mouseplane):NEXT
start.get.response:
first.talk = 0   :'used in speaking controllers commands
tts$ = ""    :'empty the string to pronounce
LOCATE 1,1
DrawText! prompt$
LOCATE 1,68
DrawText! "CMD:"+SPACE$(20)
LOCATE 1,72
GOSUB construct.response
IF mouseans$ = "" GOTO cmd.all.done    :'no action taken
IF LEFT$(mouseans$,1) <> "#" THEN GOSUB parse.command: GOTO start.cmd.decode
REM  # means debug command, always 2 letters followed by a number
num.cmds = 1
comd$(1) = LEFT$(mouseans$,3)
cmd.value(1) = VAL(RIGHT$(mouseans$,LEN(mouseans$)-3))

start.cmd.decode:

FOR cmd = 1 TO num.cmds

  function$ = comd$(cmd)
  input.value = cmd.value(cmd)

  IF function$ = "." GOTO asgd.equal.req    :'make assigned = requested A/S/H
  IF function$ = "0" GOTO handoff.action   :'accept/initiate handoff
  IF function$ = "5" GOTO keypad.hdg    :'heading on keypad
  IF function$ >= "1" AND function$ <= "9" GOTO modify.id.block  :'change
position of ID block
  IF function$ = "+" OR function$ = "-" OR function$ = "*" GOTO leader.line
:'adjust leader line
  IF function$ = "=" GOTO toggle.holding.pattern
  IF UCASE$(function$) = "A" GOTO modify.alt
  IF UCASE$(function$) = "B" GOTO set.alt.and.speed
  IF UCASE$(function$) = "C" GOTO toggle.clr.approach
  IF UCASE$(function$) = "D" GOTO divert.plane
  IF UCASE$(function$) = "H" GOTO modify.hdg
  IF UCASE$(function$) = "P" GOTO toggle.project
  IF UCASE$(function$) = "S" GOTO modify.speed
  IF UCASE$(function$) = "V" GOTO head.for.vor
  IF UCASE$(function$) = "W" GOTO toggle.warea
  IF UCASE$(function$) = "X" GOTO toggle.advisory

  'debug commands

  IF UCASE$(function$) = "#ST" GOTO dbgcmd.status
  IF UCASE$(function$) = "#XC" GOTO dbgcmd.xcoord
  IF UCASE$(function$) = "#YC" GOTO dbgcmd.ycoord
  IF UCASE$(function$) = "#AL" GOTO dbgcmd.carrier
  IF UCASE$(function$) = "#FL" GOTO dbgcmd.flight
  IF UCASE$(function$) = "#CA" GOTO dbgcmd.curalt
  IF UCASE$(function$) = "#CS" GOTO dbgcmd.curspd
  IF UCASE$(function$) = "#CH" GOTO dbgcmd.curhdg
  IF UCASE$(function$) = "#RA" GOTO dbgcmd.reqalt
  IF UCASE$(function$) = "#RS" GOTO dbgcmd.reqspd
  IF UCASE$(function$) = "#RH" GOTO dbgcmd.reqhdg
  IF UCASE$(function$) = "#VC" GOTO dbgcmd.voice
  IF UCASE$(function$) = "#HS" GOTO dbgcmd.hand

  ecode = 0
  GOSUB speak.invalid.msg: GOTO start.get.response

  asgd.equal.req:    :'make assigned A/S/H = requested A/S/H
  plane.data(10,mouseplane) = plane.data(13,mouseplane)
  plane.data(11,mouseplane) =  plane.data(14,mouseplane)
  IF plane.data(15,mouseplane) < 0 THEN plane.data(12,mouseplane) =
jetway.check.data(ABS(plane.data(15,mouseplane))) ELSE plane.data(12,mouseplane)
= plane.data(15,mouseplane)
  mcode = 1: GOSUB ctalk
  GOTO end.get.response

  modify.alt:
  ecode = 1
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)>-
9) AND (input.value < 40 OR input.value > 400) THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)>-
9) AND (input.value < 15 OR input.value > 150) THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 400) THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 150) THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  plane.data(10,mouseplane) = input.value
  mcode = 2: GOSUB ctalk
  GOTO end.get.response
  modify.speed:
  ecode = 2
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)>-
9) AND (input.value < 20 OR input.value > 70) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)>-
9) AND (input.value < 10 OR input.value > 20) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 70) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 20) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  plane.data(11,mouseplane) =  input.value
  mcode = 3: GOSUB ctalk
  GOTO end.get.response

  modify.hdg:
  ecode = 3
  IF input.value < 0 OR input.value > 359 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  plane.data(12,mouseplane) =  input.value
  mcode = 4: GOSUB ctalk
  GOTO end.get.response

  keypad.hdg:
  ecode = 14
  IF input.value < 1 OR input.value > 9 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF input.value = 5 THEN GOSUB speak.invalid.msg: GOTO start.get.response
  plane.data(12,mouseplane) =  keypad.hdgs(input.value)
  mcode = 4: GOSUB ctalk
  GOTO end.get.response

  toggle.clr.approach:
  ecode = 12
  IF plane.data(15,mouseplane) > -9 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF plane.data(22,mouseplane) = 1 THEN plane.data(22,mouseplane) = 0 ELSE
plane.data(22,mouseplane) = 1
  mcode = 9: GOSUB ctalk
  IF plane.data(22,mouseplane) = 0 GOTO end.get.response
  plane.data(12,mouseplane) = plane.data(15,mouseplane) + 8    :'VOR heading
  GOTO end.get.response

  handoff.action:
  ecode = 13
  IF plane.data(23,mouseplane) = 2 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF plane.data(23,mouseplane) = 0 THEN mcode =10: GOSUB ctalk:
plane.data(23,mouseplane) = 1: GOTO end.get.response
  IF plane.data(15,mouseplane) < -8 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  direc = plane.data(15,mouseplane)  :'requested heading
  currx = plane.data(1,mouseplane): curry = plane.data(2,mouseplane)
  p1=handoff.inset
  IF direc >= 0 THEN hdgdir = hdg.direction(1,direc) ELSE hdgdir = 0

  IF hdgdir = 1 AND handoff.status(1) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 5 AND handoff.status(2) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 3 AND handoff.status(3) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 7 AND handoff.status(4) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 2 AND (handoff.status(1) = 1 AND handoff.status(3) = 1) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 4 AND (handoff.status(3) = 1 AND handoff.status(2) = 1) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 6 AND (handoff.status(2) = 1 AND handoff.status(4) = 1) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 8 AND (handoff.status(1) = 1 AND handoff.status(4) = 1) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -1 AND handoff.status(1) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -2 AND handoff.status(2) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -3 AND handoff.status(3) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -4 AND handoff.status(4) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -5 AND aw4ep$ = "N" AND handoff.status(1) = 1 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -5 AND aw4ep$ = "E" AND handoff.status(3) = 1 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -6 AND handoff.status(4) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -7 AND aw13ep$ = "S" AND handoff.status(2) = 1 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -7 AND aw13ep$ = "E" AND handoff.status(3) = 1 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -8 AND handoff.status(1) = 1 THEN GOSUB speak.invalid.msg: GOTO
start.get.response

  IF hdgdir = 1 AND (curry<=p1) GOTO ok.hdg
  IF hdgdir = 2 AND ((curry<=p1) OR (currx>=mac.screen.width-4-p1)) GOTO ok.hdg
  IF hdgdir = 3 AND (currx>=mac.screen.width-4-p1) GOTO ok.hdg
  IF hdgdir = 4 AND ((currx>=mac.screen.width-4-p1) OR
(curry>=mac.screen.height-26-p1)) GOTO ok.hdg
  IF hdgdir = 5 AND (curry>=mac.screen.height-26-p1) GOTO ok.hdg
  IF hdgdir = 6 AND ((curry>=mac.screen.height-26-p1) OR (currx<=p1)) GOTO
ok.hdg
  IF hdgdir = 7 AND (currx<=p1) GOTO ok.hdg
  IF hdgdir = 8 AND ((curry<=p1) OR (currx<=p1)) GOTO ok.hdg

  IF direc = -1 AND (curry<=p1) GOTO ok.hdg
  IF direc = -2 AND (curry>=mac.screen.height-26-p1) GOTO ok.hdg
  IF direc = -3 AND (currx>=mac.screen.width-4-p1) GOTO ok.hdg
  IF direc = -4 AND (currx<=p1) GOTO ok.hdg
  IF direc = -5 AND aw4ep$ = "N" AND (curry<=p1) GOTO ok.hdg
  IF direc = -5 AND aw4ep$ = "E" AND (currx>=mac.screen.width-4-p1) GOTO ok.hdg
  IF direc = -6 AND (currx<=p1) GOTO ok.hdg
  IF direc = -7 AND aw13ep$ = "S" AND (curry>=mac.screen.height-26-p1) GOTO
ok.hdg
  IF direc = -7 AND aw13ep$ = "E" AND (currx>=mac.screen.width-4-p1) GOTO ok.hdg
  IF direc = -8 AND (curry<=p1) GOTO ok.hdg

  GOSUB speak.invalid.msg: GOTO start.get.response
  ok.hdg:
  mcode = 10: GOSUB ctalk
  plane.data(23,mouseplane) = 2
  GOTO end.get.response

  toggle.project:
  ecode = 4
  IF input.value > 15 THEN GOSUB speak.invalid.msg: GOTO start.get.response
  IF plane.data(16,mouseplane) <> 0 AND input.value = 0 THEN
plane.data(16,mouseplane) = 0: GOTO end.get.response
  IF plane.data(16,mouseplane) <> 0 THEN plane.data(16,mouseplane) =
input.value: GOTO end.get.response
  IF input.value = 0 THEN plane.data(16,mouseplane) = -1 ELSE
plane.data(16,mouseplane) = input.value
  GOTO end.get.response

  toggle.warea:
  ecode = 5
  IF input.value > 50 THEN GOSUB speak.invalid.msg: GOTO start.get.response
  IF plane.data(18,mouseplane) <> 0 AND input.value = 0 THEN
plane.data(18,mouseplane) = 0: GOTO end.get.response
  IF plane.data(18,mouseplane) <> 0 THEN plane.data(18,mouseplane) = input.value
+ 1: GOTO end.get.response
  IF input.value = 0 THEN plane.data(18,mouseplane) = -1 ELSE
plane.data(18,mouseplane) = input.value + 1
  GOTO end.get.response

  toggle.advisory:
  IF plane.data(21,mouseplane) = 0 THEN plane.data(21,mouseplane) = 1 ELSE
plane.data(21,mouseplane) = 0
  GOTO end.get.response

  divert.plane:
  ecode = 11
  IF input.value <> 36 AND input.value <> 18 AND input.value <> 27 AND
input.value <> 9  THEN GOSUB speak.invalid.msg: GOTO start.get.response
  direc = plane.data(15,mouseplane)

  IF direc >= 0 THEN hdgdir = hdg.direction(1,direc) ELSE hdgdir = 0
  IF hdgdir = 1 AND handoff.status(1) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 5 AND handoff.status(2) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 3 AND handoff.status(3) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
```

```
  IF hdgdir = 7 AND handoff.status(4) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF hdgdir = 2 AND (handoff.status(1) = 0 OR handoff.status(3) = 0) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 4 AND (handoff.status(3) = 0 OR handoff.status(2) = 0) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 6 AND (handoff.status(2) = 0 OR handoff.status(4) = 0) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF hdgdir = 8 AND (handoff.status(1) = 0 OR handoff.status(4) = 0) THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -1 AND handoff.status(1) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -2 AND handoff.status(2) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -3 AND handoff.status(3) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -4 AND handoff.status(4) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -5 AND aw4ep$ = "N" AND handoff.status(1) = 0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -5 AND aw4ep$ = "E" AND handoff.status(3) = 0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -6 AND handoff.status(4) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF direc = -7 AND aw13ep$ = "S" AND handoff.status(2) = 0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -7 AND aw13ep$ = "E" AND handoff.status(3) = 0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF direc = -8 AND handoff.status(1) = 0 THEN GOSUB speak.invalid.msg: GOTO
start.get.response

  IF direc < -8 AND airport.status(ABS(direc)-8) = 0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF input.value = 36 AND airport.status(1) = 1 THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF input.value = 18 AND airport.status(2) = 1 THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF input.value = 27 AND airport.status(3) = 1 THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF input.value = 9 AND airport.status(4) = 1 THEN GOSUB speak.invalid.msg:
GOTO start.get.response
  IF input.value = 36 THEN plane.data(15,mouseplane) = -9
  IF input.value = 18 THEN plane.data(15,mouseplane) = -10
  IF input.value = 27 THEN plane.data(15,mouseplane) = -11
  IF input.value = 9 THEN plane.data(15,mouseplane) = -12
  plane.data(13,mouseplane) = 10: plane.data(14,mouseplane) = 10
  mcode = 8: GOSUB ctalk
  GOTO end.get.response

  modify.id.block:
  ecode = 6
  IF function$ = "5" THEN GOSUB speak.invalid.msg: GOTO start.get.response
  plane.data(9,mouseplane) = idblkpos(VAL(function$))
  GOTO end.get.response
  leader.line:
  ecode = 7
  IF function$ = "*" THEN plane.data(20,mouseplane) = 0: GOTO end.get.response
  IF input.value = 0 THEN input.value = 1
  IF function$ = "+" AND plane.data(20,mouseplane)+input.value > 15 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF function$ = "-" AND plane.data(20,mouseplane)-input.value <  0 THEN GOSUB
speak.invalid.msg: GOTO start.get.response
  IF function$ = "+" THEN plane.data(20,mouseplane) = plane.data(20,mouseplane)
+ input.value ELSE plane.data(20,mouseplane) = plane.data(20,mouseplane) -
input.value
  GOTO end.get.response

  toggle.holding.pattern:
  ecode = 8
  IF input.value > 50 THEN GOSUB speak.invalid.msg: GOTO start.get.response
  IF plane.data(17,mouseplane) <> 0 AND input.value = 0 THEN
plane.data(17,mouseplane) = 0: GOTO hold.1
  IF plane.data(17,mouseplane) <> 0 THEN plane.data(17,mouseplane) =
input.value: GOTO hold.1
  IF input.value = 0 THEN plane.data(17,mouseplane) = -1 ELSE
plane.data(17,mouseplane) = input.value
  hold.1:  mcode = 5: GOSUB ctalk
  GOTO end.get.response

  head.for.vor:
  ecode = 9
  IF input.value <> 36 AND input.value <> 18 AND input.value <> 27 AND
input.value <> 9  THEN GOSUB speak.invalid.msg: GOTO start.get.response
  IF input.value = 36 THEN plane.data(12,mouseplane) = -1
  IF input.value = 18 THEN plane.data(12,mouseplane) = -2
  IF input.value = 27 THEN plane.data(12,mouseplane) = -3
  IF input.value = 9 THEN plane.data(12,mouseplane) = -4
  mcode = 6: GOSUB ctalk
  GOTO end.get.response

  set.alt.and.speed:
  ecode = 10
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)>-
9) AND (input.value < 40 OR input.value > 70) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)>-
9) AND (input.value < 15 OR input.value > 20) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 1 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 70) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  IF (carrier.size(plane.data(3,mouseplane)) = 0 AND plane.data(15,mouseplane)<-
8) AND (input.value < 10 OR input.value > 20) THEN GOSUB speak.invalid.msg: GOTO
start.get.response
  plane.data(10,mouseplane) =  input.value
  plane.data(11,mouseplane) =  input.value
  mcode = 7: GOSUB ctalk
  GOTO end.get.response
```

```
dbgcmd.status:
plane.data(0,mouseplane) = input.value
IF plane.data(0,mouseplane) <> 0 GOTO end.get.response
FOR jff = 0 TO plane.params: plane.data(jff,mouseplane) = 0: NEXT
GOTO end.get.response

dbgcmd.xcoord:
plane.data(1,mouseplane) = input.value
GOTO end.get.response

dbgcmd.ycoord:
plane.data(2,mouseplane) = input.value
GOTO end.get.response

dbgcmd.curalt:
plane.data(5,mouseplane) = input.value
GOTO end.get.response

dbgcmd.curspd:
plane.data(6,mouseplane) = input.value
GOTO end.get.response

dbgcmd.curhdg:
plane.data(7,mouseplane) = input.value
GOTO end.get.response

dbgcmd.reqalt:
plane.data(13,mouseplane) = input.value
GOTO end.get.response

dbgcmd.reqspd:
plane.data(14,mouseplane) = input.value
GOTO end.get.response

dbgcmd.reqhdg:
plane.data(15,mouseplane) = input.value
GOTO end.get.response

dbgcmd.voice:
plane.data(19,mouseplane) = input.value
GOTO end.get.response

dbgcmd.hand:
plane.data(23,mouseplane) = input.value
GOTO end.get.response

dbgcmd.carrier:
plane.data(3,mouseplane) = input.value
GOTO end.get.response

dbgcmd.flight:
plane.data(4,mouseplane) = input.value
GOTO end.get.response

end.get.response:
```

```
NEXT

cmd.all.done:
IF first.talk <> 0 THEN voice.type = -1: GOSUB speakit

WINDOW 1: GOSUB redraw.one.plane

RETURN

REM  **********************************************
handle.button:   'user has pressed a button in the options screen

button.num = DIALOG(1)

GOSUB handle.editfield   :'in case any edit field was updated

IF button.num < 4 THEN GOSUB speak.text: GOTO handle.button.exit
IF button.num < 17 THEN GOSUB update.airline.size: GOTO handle.button.exit
IF button.num < 30 THEN GOSUB speak.text: GOTO handle.button.exit
IF button.num = 30 THEN GOSUB save.options: GOTO handle.button.exit
IF button.num = 31 THEN xwait=0: GOTO handle.button.exit
IF button.num < 42 THEN GOSUB update.modes: GOTO handle.button.exit
IF button.num < 58 THEN GOSUB update.levels: GOTO handle.button.exit
IF button.num < 60 THEN GOSUB update.counts: GOTO handle.button.exit

handle.button.exit:

RETURN

REM  **********************************************
handle.editfield:   'user has entered an edit field in the options screen

center.name$ = EDIT$(1)
airport$(1) = EDIT$(2): airport$(2)=airport$(1)
airport$(3) = EDIT$(3): airport$(4)=airport$(3)
FOR i = 0 TO 12
  carrier$(i) = LEFT$(EDIT$(i+4),2)
  carrier.names$(i) = EDIT$(i+17)
NEXT
IF VAL(EDIT$(30)) > max.planes THEN params(9) = max.planes ELSE
params(9)=VAL(EDIT$(30))
params(11)=VAL(EDIT$(31))
IF VAL(EDIT$(32)) > 60 THEN params(13) = 60 ELSE params(13)= VAL(EDIT$(32))
IF VAL(EDIT$(33)) > 100 THEN adv.cycles = 100 ELSE adv.cycles = VAL(EDIT$(33))
IF UCASE$(LEFT$(EDIT$(34),1)) = "D" THEN debug.switch = -debug.switch

RETURN

REM  **********************************************
handle.handoff:   'see if you want to close an airport or handoffs

IF INT(RND*250*params(5)) <> 100 THEN RETURN
tts$ = center.name$ + " center, "
gr.color = 30  :'white for erasing lines
```

```
x = INT(RND*8)+1
ON x GOTO
toggle.handoff1,toggle.handoff2,toggle.handoff3,toggle.handoff4,toggle.airport1r
1,toggle.airport1r2,toggle.airport2r1,toggle.airport2r2
RETURN

toggle.handoff1:
IF handoff.status(1) = 0 THEN tts$ = tts$ + direction$(1)+hand.msg1$:
handoff.status(1) = 1: GOTO hand.exit
tts$ = tts$ + hand.msg2$+direction$(1)
handoff.status(1) = 0
LINE (0,0)-(mac.screen.width-4,7),gr.color,BF: GOSUB draw.airways
GOTO hand.exit

toggle.handoff2:
IF handoff.status(2) = 0 THEN tts$ = tts$ + direction$(2)+hand.msg1$:
handoff.status(2) = 1: GOTO hand.exit
tts$ = tts$ + hand.msg2$+direction$(2)
handoff.status(2) = 0
LINE (0,mac.screen.height-35)-(mac.screen.width-4,mac.screen.height-
26),gr.color,BF: GOSUB draw.airways
GOTO hand.exit

toggle.handoff3:
IF handoff.status(3) = 0 THEN tts$ = tts$ + direction$(3)+hand.msg1$:
handoff.status(3) = 1: GOTO hand.exit
tts$ = tts$ + hand.msg2$+direction$(3)
handoff.status(3) = 0
LINE (mac.screen.width-12,0)-(mac.screen.width-4,mac.screen.height-
26),gr.color,BF: GOSUB draw.airways
GOTO hand.exit

toggle.handoff4:
IF handoff.status(4) = 0 THEN tts$ = tts$ + direction$(4)+hand.msg1$:
handoff.status(4) = 1: GOTO hand.exit
tts$ = tts$ + hand.msg2$+direction$(4)
handoff.status(4) = 0
LINE (0,0)-(7,mac.screen.height-26),gr.color,BF: GOSUB draw.airways
GOTO hand.exit

toggle.airport1r1:
tts$ = tts$ + "runway thirty six at " + airport$(1)
IF airport.status(1) = 0 THEN tts$ = tts$ + close.msg1$: airport.status(1) = 1:
GOTO hand.exit
tts$ = tts$ + close.msg2$
airport.status(1) = 0
LINE (ap1x-2,ap1y+33)-(ap1x+4,ap1y+35),gr.color,BF
GOTO hand.exit

toggle.airport1r2:
tts$ = tts$ + "runway eighteen at " + airport$(2)
IF airport.status(2) = 0 THEN tts$ = tts$ + close.msg1$: airport.status(2) = 1:
GOTO hand.exit
tts$ = tts$ + close.msg2$
airport.status(2) = 0
```

```
LINE (ap1x-2,ap1y-5)-(ap1x+4,ap1y-3),gr.color,BF
GOTO hand.exit

toggle.airport2r1:
tts$ = tts$ + "runway twenty seven at " + airport$(3)
IF airport.status(3) = 0 THEN tts$ = tts$ + close.msg1$: airport.status(3) = 1:
GOTO hand.exit
tts$ = tts$ + close.msg2$
airport.status(3) = 0
LINE (ap2x+33,ap2y-2)-(ap2x+35,ap2y+4),gr.color,BF
GOTO hand.exit

toggle.airport2r2:
tts$ = tts$ + "runway nine at " + airport$(4)
IF airport.status(4) = 0 THEN tts$ = tts$ + close.msg1$: airport.status(4) = 1:
GOTO hand.exit
tts$ = tts$ + close.msg2$
airport.status(4) = 0
LINE (ap2x-5,ap2y-2)-(ap2x-3,ap2y+4),gr.color,BF
GOTO hand.exit

hand.exit:
voice.type = -1
GOSUB speakit
GOSUB draw.airways

RETURN

REM  **********************************************
inverse.video:   'reverse the color of data blocks, for emphasis

'  index ndx is the plane to be reversed

x = plane.data(1,ndx): y = plane.data(2,ndx): z = plane.data(9,ndx): zz =
plane.data(20,ndx)

rectangle(0) = y+id.block.data(6,z,zz)-9
rectangle(1) = x+id.block.data(5,z,zz)-2
rectangle(2) = y+id.block.data(6,z,zz)+10
rectangle(3) = x+id.block.data(5,z,zz)+36

CALL INVERTRECT (VARPTR(rectangle(0)))

RETURN

REM  **********************************************
landing.check:   'see if a plane destined for a runway has landed, and give msg

FOR lck = 0 TO max.index
  IF plane.data(0,lck) <> 1 GOTO lck.exit    :'inactive or unidentified, ignore
it
  IF plane.data(15,lck) >= -8 GOTO lck.exit    :'not on runway heading
  rcp = ABS(plane.data(15,lck)) - 8
  IF plane.data(1,lck) < runway.ck.params(1,rcp) - 5 OR plane.data(1,lck) >
runway.ck.params(1,rcp) + 5 GOTO lck.exit
```

```
  IF plane.data(2,lck) < runway.ck.params(2,rcp) - 5 OR plane.data(2,lck) >
runway.ck.params(2,rcp) + 5 GOTO lck.exit
  IF plane.data(7,lck) <> jetway.check.data(ABS(plane.data(15,lck))) GOTO
lck.exit
  REM the plane is at the runway and heading, make sure a/s are correct
  IF plane.data(5,lck) > 15 GOTO not.configured
  IF plane.data(6,lck) > 15 GOTO not.configured
  REM plane has landed
  REM he crashed if the runway was closed
  IF airport.status(rcp) = 0 GOTO landed.ok
  planes.crashed.total = planes.crashed.total + 1
  tts$ = carrier.names$(plane.data(3,lck))+STR$(plane.data(4,lck))+"  HAS
CRASHED AT "+airport$(rcp)
  tts$ = tts$ + " -- CALL NTSB  AT ONCE"
  GOTO speak.landing.msg
  landed.ok:
  planes.landed.successfully = planes.landed.successfully + 1
  t$ = depart.msg$(INT(4*RND)+1)
  tts$ = carrier.names$(plane.data(3,lck))+STR$(plane.data(4,lck))+"  HAS LANDED
AT "+airport$(rcp)+t$
  speak.landing.msg:
  voice.type = lck
  GOSUB speakit
  k = lck
  GOSUB paint.it.white
  FOR lck1 = 0 TO plane.params: plane.data(lck1,lck) = 0: NEXT    :'erase plane
data
  GOTO lck.exit
  not.configured:    :'plane is at improper a/s/h for landing
  tts$ = carrier.names$(plane.data(3,lck))+STR$(plane.data(4,lck))+"  IS NOT
CONFIGURED FOR LANDING"
  voice.type = lck
  GOSUB speakit

  lck.exit:
NEXT
RETURN

REM  **********************************************
make.new.plane:    'to create everything for a new plane

IF j > max.index THEN max.index = j
planes.activated = planes.activated + 1
plane.data(0,j) = 1  :' active plane
plane.data(1,j) = INT((mac.screen.width-47)*RND)+15  :'x coordinate
plane.data(2,j) = INT((mac.screen.height-72)*RND)+15  :'y coordinate
plane.data(3,j) = INT(13*RND)  :'airline
plane.data(4,j) = INT(999*RND) + 1  :'flight number
IF carrier.size(plane.data(3,j)) = 1 THEN plane.data(5,j) = INT(360*RND)+40
:'altitude
IF carrier.size(plane.data(3,j)) = 0 THEN plane.data(5,j) = INT(135*RND)+15
:'altitude
IF (plane.data(1,j)>ap2x-61 AND plane.data(1,j)<ap2x+91) AND
(plane.data(2,j)>ap2y-40 AND plane.data(2,j)<ap2y+42) AND (plane.data(5,j) < 55)
THEN plane.data(5,j) = 55
```

```
IF (plane.data(1,j)>ap1x-40 AND plane.data(1,j)<ap1x+42) AND
(plane.data(2,j)>ap1y-61 AND plane.data(2,j)<ap1y+91) AND (plane.data(5,j) < 55)
THEN plane.data(5,j) = 55
IF carrier.size(plane.data(3,j)) = 1 THEN plane.data(6,j) = INT(50*RND)+20
:'speed
IF carrier.size(plane.data(3,j)) = 0 THEN plane.data(6,j) = INT(10*RND)+10
:'speed
plane.data(7,j) = INT(359*RND)  :'heading
plane.data(8,j) = 0  :'attitude, N/A for first cycle
plane.data(9,j) = preferred.db.pos(1,hdg.direction(1,plane.data(7,j)))  :'data
block position
IF carrier.size(plane.data(3,j)) = 1 THEN plane.data(10,j) = INT(360*RND)+40
:'assigned altitude
IF carrier.size(plane.data(3,j)) = 0 THEN plane.data(10,j) = INT(135*RND)+15
IF carrier.size(plane.data(3,j)) = 1 THEN plane.data(11,j) = INT(50*RND)+20
:'assigned speed
IF carrier.size(plane.data(3,j)) = 0 THEN plane.data(11,j) = INT(10*RND)+10
plane.data(12,j) = plane.data(7,j)  :'assigned heading = current heading
plane.data(13,j) = plane.data(10,j)  :'requested alt. = assigned alt.
plane.data(14,j) = plane.data(11,j)  :'requested speed = assigned speed
IF params(6) = 1 GOTO simtype.mix
IF params(6) = 2 GOTO simtype.enroute
IF params(6) = 3 GOTO simtype.approach
simtype.mix:
i = INT(2*RND)  :'req. heading = 50% a heading, 50% a jetway/runway
IF i > 0 OR max.planes.handled = 0 THEN plane.data(15,j) = plane.data(12,j):
GOTO simtype.exit  :'heading
plane.data(15,j) = -(INT(12*RND)+1)  :'jetways/runways are negative numbers
IF params(1) = 1 AND plane.data(15,j) = -5 THEN plane.data(15,j) = -1
IF params(1) = 1 AND plane.data(15,j) = -6 THEN plane.data(15,j) = -2
IF params(1) = 1 AND plane.data(15,j) = -7 THEN plane.data(15,j) = -3
IF params(1) = 1 AND plane.data(15,j) = -8 THEN plane.data(15,j) = -4
IF plane.data(15,j) <= -9 AND plane.data(5,j) > 60 THEN plane.data(5,j) = 60
IF plane.data(15,j) <= -9 THEN plane.data(10,j) = 10: plane.data(11,j) = 10:
plane.data(13,j) = 10: plane.data(14,j) = 10
GOTO simtype.exit
simtype.enroute:
i = INT(2*RND)  :'req. heading = 50% a heading, 50% a jetway/runway
IF i > 0 OR max.planes.handled = 0 THEN plane.data(15,j) = plane.data(12,j):
GOTO simtype.exit  :'heading
plane.data(15,j) = -(INT(8*RND)+1)  :'jetways/runways are negative numbers
IF params(1) = 1 THEN plane.data(15,j) = -(INT(4*RND)+1)  :'no J4/22/13/31 in
demo mode
GOTO simtype.exit
simtype.approach:    :'all have a runway heading
plane.data(15,j) = -(INT(4*RND)+1+8)
IF plane.data(15,j) <= -9 AND plane.data(5,j) > 60 THEN plane.data(5,j) = 60
IF plane.data(15,j) <= -9 THEN plane.data(10,j) = 10: plane.data(11,j) = 10:
plane.data(13,j) = 10: plane.data(14,j) = 10

simtype.exit:
plane.data(16,j) = 0    :'don't print projected course
plane.data(17,j) = 0    :'not in holding pattern
plane.data(18,j) = 0    :'don't print warning area
plane.data(19,j) = INT(100*RND)+65   :'male/female voice
plane.data(20,j) = 0    :'normal leader line length
plane.data(21,j) = 0    :'conflict advisory enabled
plane.data(22,j) = 0    :'not cleared for the approach
IF max.planes.handled = 0 THEN plane.data(23,j) = 1 ELSE plane.data(23,j) = 0
:'handoff status

RETURN

REM **********************************************
menuproc:   'go here when a menu choice is made

MOUSE OFF  :'ignore mouse presses for now
menu.num = MENU(0): menu.item = MENU(1)
IF menu.num = 1 AND menu.item = 1 THEN MENU: tts$="thank you for playing
professional air traffic controller simulaytor": voice.type = -1:GOSUB speakit:
speechoff! speechhand!: STOP
IF menu.num = 1 AND menu.item = 2 THEN MENU: rerun.sw = 1: delay.switch = 1:
GOTO menuexit
IF menu.num = 1 AND menu.item = 3 THEN MENU: GOSUB display.options: GOTO
menuexit
IF menu.num = 1 AND menu.item = 4 THEN MENU: rerun.sw1 = 1: delay.switch = 1:
speechoff! speechhand!: GOTO menuexit

IF menu.num = 2 AND menu.item = 1 THEN MENU: xwait = 0: WINDOW 1: GOSUB
paint.with.clear: GOTO menuexit
IF menu.num = 2 AND menu.item = 2 THEN MENU: GOSUB display.status: GOTO menuexit
IF menu.num = 2 AND menu.item = 3 THEN MENU: GOSUB display.score: GOTO menuexit
IF menu.num = 2 AND menu.item = 4 THEN MENU: GOSUB display.ref.screen: GOTO
menuexit
IF menu.num = 2 AND menu.item = 5 THEN MENU: GOSUB display.command.help: GOTO
menuexit

IF menu.num = 3 THEN MENU: GOSUB selective.display: GOTO menuexit

menuexit:
MOUSE ON   :'reenable mouse activity
RETURN

REM **********************************************
mouseproc:   'go here when a mouse press is made

IF xwait = 1 THEN xwait = 0: WINDOW 1: GOSUB paint.with.clear: RETURN
mcnt:
mousez = MOUSE(0)
mousex = MOUSE(3)         :'determine beginning coords of mouse pointer
mousey = MOUSE(4)
mouseex = MOUSE(5)         :'determine ending coords of mouse pointer
mouseey = MOUSE(6)
IF mousez < 0 GOTO mcnt    :'mouse is still down

FOR mousez = 0 TO max.index    :'find out which plane he wants to update (+- 8
pixels)
  IF plane.data(0,mousez) <> 1 GOTO ck.nxt.possbl.plane  :'inactive or unident
flag
  'see if clicked on plane dot
  xxx1 = plane.data(1,mousez) - 5
  yyy1 = plane.data(2,mousez) - 5
  xxx2 = plane.data(1,mousez) + 5
  yyy2 = plane.data(2,mousez) + 5
  IF (mousex >= xxx1 AND mousex <= xxx2) AND (mousey >= yyy1 AND mousey <= yyy2)
GOTO mousefound

  'see if clicked on data block
  xxx1 = plane.data(1,mousez) +
coord.ck.params(1,plane.data(9,mousez),plane.data(20,mousez))
  yyy1 = plane.data(2,mousez) +
coord.ck.params(2,plane.data(9,mousez),plane.data(20,mousez))
  xxx2 = plane.data(1,mousez) +
coord.ck.params(3,plane.data(9,mousez),plane.data(20,mousez))
  yyy2 = plane.data(2,mousez) +
coord.ck.params(4,plane.data(9,mousez),plane.data(20,mousez))
  IF (mousex >= xxx1 AND mousex <= xxx2) AND (mousey >= yyy1 AND mousey <= yyy2)
GOTO mousefound
  ck.nxt.possbl.plane:
NEXT
delay.switch = 1    :'accelerate the simulation
GOTO mousexit

mousefound:
IF ABS(mousex-mouseex) > 5 OR ABS(mousey-mouseey) > 5 THEN GOSUB drag.db: GOTO
mousexit

GOSUB build.cmd.prompt
GOSUB get.response

mousexit:
RETURN

REM  ***********************************************
move.db:   'move the data block based upon a mouse drag

k = mousez: GOSUB paint.it.white
plane.data(9,mousez) = t1
plane.data(20,mousez) = t2
k = mousez: GOSUB paint.it.black

RETURN

REM  ***********************************************
paint.it.black:   'draw one plane on screen

CALL TEXTMODE (1):  gr.color = 33
GOSUB paint.wo.clear
RETURN

REM  ***********************************************
paint.it.white:   'erase one plane from screen

CALL TEXTMODE (3):  gr.color = 30
GOSUB paint.wo.clear
GOSUB draw.airways
RETURN

REM  **********************************************
paint.update.screen:   'paint plane white, update vals, paint black

FOR k = 0 TO max.index
  IF plane.data(0,k) = 0 GOTO end.recalcx   :'it is inactive, so don't update
  CALL TEXTMODE (3): gr.color = 30
  GOSUB paint.wo.clear
  GOSUB update.current.values
  CALL TEXTMODE (1): gr.color = 33
  GOSUB paint.wo.clear
  end.recalcx:
NEXT
GOSUB draw.airways
MOUSE ON: MENU ON   :'respond to mouse or menu activity while updating screen
MOUSE STOP: MENU STOP
RETURN

REM  **********************************************
paint.with.clear:    'clear the screen and repaint everything

CLS
CALL TEXTMODE (1):  gr.color = 33
GOSUB draw.airways
FOR k = 0 TO max.index
  IF plane.data(0,k) <> 0 THEN GOSUB paint.wo.clear
NEXT
RETURN

REM  **********************************************
paint.wo.clear:   'paint one plane on the screen (index k)

x = plane.data(1,k)
y = plane.data(2,k)
LINE (x-1,y-1)-(x+1,y+1),gr.color,B
IF plane.data(0,k) = 2 GOTO pwoc.exit  :'unidentified aircraft
hdg = plane.data(7,k): hdg1 = plane.data(15,k)
hdgdir = hdg.direction(1,hdg)
IF hdg1 >= 0 THEN hdgdir1 = hdg.direction(1,hdg1) ELSE hdgdir1 =
arrow.hdg(ABS(hdg1))
LINE (x,y)-(x+hdg.direction(2,hdg),y+hdg.direction(3,hdg)),gr.color
ipd = plane.data(9,k)
xx1 = x + id.block.data(1,ipd,plane.data(20,k))
yy1 = y + id.block.data(2,ipd,plane.data(20,k))
xx2 = x + id.block.data(3,ipd,plane.data(20,k))
yy2 = y + id.block.data(4,ipd,plane.data(20,k))
xx3 = x + id.block.data(5,ipd,plane.data(20,k))
yy3 = y + id.block.data(6,ipd,plane.data(20,k))
xx4 = x + id.block.data(7,ipd,plane.data(20,k))
yy4 = y + id.block.data(8,ipd,plane.data(20,k))
LINE (xx1,yy1)-(xx2,yy2),gr.color
IF params(3) = 1 GOTO pwoc.exit
```

```
LINE (x+arrow.data(1,hdgdir1),y+arrow.data(2,hdgdir1))-
(x+arrow.data(3,hdgdir1),y+arrow.data(4,hdgdir1)),gr.color
LINE (x+arrow.data(1,hdgdir1),y+arrow.data(2,hdgdir1))-
(x+arrow.data(5,hdgdir1),y+arrow.data(6,hdgdir1)),gr.color
CALL MOVETO (xx3,yy3)
IF plane.data(21,k) = 1 THEN CALL TEXTFACE (4)
DrawText! carrier$(plane.data(3,k))+MID$(STR$(plane.data(4,k)),2,4)
IF plane.data(23,k) = 0 THEN DrawText! "X": GOTO noastskprt
IF plane.data(22,k) <> 0 THEN DrawText! "C": GOTO noastskprt
IF plane.data(17,k) <> 0 THEN DrawText! "=": GOTO noastskprt
IF plane.data(15,k) < -8 THEN DrawText! "R": GOTO noastskprt
IF plane.data(10,k) <> plane.data(13,k) GOTO print.astsk
IF plane.data(11,k) <> plane.data(14,k) GOTO print.astsk
IF plane.data(15,k) >= 0 AND plane.data(12,k) <> plane.data(15,k) GOTO
print.astsk
IF plane.data(23,k) = 2 THEN DrawText! "H": GOTO noastskprt
IF plane.data(15,k) >= 0 GOTO noastskprt
DrawText! "J"
GOTO noastskprt
print.astsk: DrawText! "*"    :'print * if assigned a/s/h not = requested a/s/h
noastskprt:
CALL MOVETO (xx4,yy4)
IF dswt.fdb = 1 THEN DrawText!
RIGHT$("00"+MID$(STR$(plane.data(5,k)),2,3),3)+asc.symbol$(plane.data(8,k)+2)+RI
GHT$("0"+MID$(STR$(plane.data(6,k)),2,2),2)
IF plane.data(21,k) = 1 THEN CALL TEXTFACE (0)
IF plane.data(18,k) <> 0 THEN CIRCLE (x,y),30,gr.color
IF plane.data(16,k) = 0 GOTO pwoc.exit
IF plane.data(16,k) > 0 GOTO proj.part.line
IF hdgdir = 1 THEN LINE (x,y) - (x,0),gr.color
IF hdgdir = 2 THEN LINE (x,y) - (x+y,0),gr.color
IF hdgdir = 3 THEN LINE (x,y) - (mac.screen.width-4,y),gr.color
IF hdgdir = 4 THEN LINE (x,y) - (mac.screen.width-4,mac.screen.width-4-
x+y),gr.color
IF hdgdir = 5 THEN LINE (x,y) - (x,mac.screen.height-26),gr.color
IF hdgdir = 6 THEN LINE (x,y) - (0,x+y),gr.color
IF hdgdir = 7 THEN LINE (x,y) - (0,y),gr.color
IF hdgdir = 8 THEN LINE (x,y) - (x-y,0),gr.color
GOTO pwoc.exit
proj.part.line:
px = plane.data(1,k): py = plane.data(2,k): ps = plane.data(6,k)
IF params(13) < 1 THEN BB = 60*plane.data(16,k) ELSE BB =
(60/params(13))*plane.data(16,k)
FOR B = 1 TO BB
  IF ps = plane.data(11,k) GOTO xzz
  IF ps < plane.data(11,k) THEN ps = ps + 1 ELSE ps = ps - 1
  xzz:
  n = speedfact(ps)
  px = px + (n*hdg.direction(4,hdg))
  py = py + (n*hdg.direction(5,hdg))
NEXT
LINE (x,y) - (px,py),gr.color

pwoc.exit:
RETURN
```

```
REM  **********************************************
parse.command:   'interpret multiple commands on one line

IF LEFT$(mouseans$,1) ="/" THEN comd$(1)="!": num.cmds = 1: GOTO parse.exit
IF RIGHT$(mouseans$,1) <> "/" THEN mouseans$ = mouseans$+"/"
num.cmds = 0
FOR cmd = 1 TO LEN(mouseans$)
  IF MID$(mouseans$,cmd,1)="/" AND MID$(mouseans$,cmd+1,1)="/" THEN
comd$(1)="!": num.cmds = 1: GOTO parse.exit
  IF MID$(mouseans$,cmd,1) = "/" THEN num.cmds = num.cmds + 1
NEXT
pntr = 1
FOR cmd = 1 TO num.cmds
  comd$(cmd) = MID$(mouseans$,pntr,1)
  epntr = INSTR(pntr,mouseans$,"/")
  IF epntr = pntr+1 THEN cmd.value(cmd)=0 ELSE
cmd.value(cmd)=VAL(MID$(mouseans$,pntr+1,epntr-pntr-1))
  pntr = epntr + 1
NEXT
parse.exit:
RETURN

REM  **********************************************
proximity.check:    'issue alert if planes are too close

IF debug.switch = -1 THEN RETURN

FOR pck.idx = 0 TO max.index-1
  IF plane.data(0,pck.idx) = 0 THEN GOTO proxck1.loop.exit  :'inactive plane
  FOR pck.idx2 = pck.idx +1 TO max.index
    IF plane.data(0,pck.idx2) = 0 THEN GOTO proxck.loop.exit  :'inactive plane
    IF plane.data(0,pck.idx) = 2 AND plane.data(0,pck.idx2) = 2 THEN GOTO
proxck.loop.exit  :'both are unidentified
    pck.x1 = plane.data(1,pck.idx): pck.y1 = plane.data(2,pck.idx): pck.alt1 =
plane.data(5,pck.idx)
    pck.x2 = plane.data(1,pck.idx2): pck.y2 = plane.data(2,pck.idx2): pck.alt2 =
plane.data(5,pck.idx2)
    xdif = ABS(pck.x1 - pck.x2): ydif = ABS(pck.y1 - pck.y2): altdif =
ABS(pck.alt1 - pck.alt2)
    IF altdif >= 10 GOTO proxck.loop.exit  :'alt separation is more than 1000
feet
    REM check positions of 2 planes & give warning if too close
    IF xdif > 30 OR ydif > 30 GOTO proxck.loop.exit  :'they aren't that close
    IF warn.test(xdif+1,ydif+1) = 1 GOTO proxck.loop.exit  :'not in circle

    REM if they are really close, then they have crashed!
    IF altdif <= 2 AND xdif <= 5 AND ydif <= 5 GOTO proc.crash  :'they crashed

    IF params(4) = 1 THEN BEEP:BEEP:BEEP
    tt1$ = carrier.names$(plane.data(3,pck.idx)) + STR$(plane.data(4,pck.idx))
    tt2$ = carrier.names$(plane.data(3,pck.idx2)) + STR$(plane.data(4,pck.idx2))
    IF plane.data(0,pck.idx) = 2 THEN tt1$ = " un identified aircraft "
    IF plane.data(0,pck.idx2) = 2 THEN tt2$ = " un identified aircraft "
    IF params(3) <> 1 THEN ndx = pck.idx: GOSUB inverse.video
    IF params(3) <> 1 THEN ndx = pck.idx2: GOSUB inverse.video
    tts$ = "KON FLICT ALERT  -----  " + tt1$ + " and " + tt2$
    voice.type = -1
    GOSUB speakit
    IF params(3) <> 1 THEN ndx = pck.idx2: GOSUB inverse.video
    IF params(3) <> 1 THEN ndx = pck.idx: GOSUB inverse.video
    planes.proximity.warnings = planes.proximity.warnings + 1  :'tally
    warn.plane(pck.idx) = 1
    warn.plane(pck.idx2) = 1
    GOTO proc.continue

    proc.crash:
    IF params(4) = 1 THEN BEEP:BEEP:BEEP:BEEP:BEEP:BEEP
    tt1$ = carrier.names$(plane.data(3,pck.idx)) + STR$(plane.data(4,pck.idx))
    tt2$ = carrier.names$(plane.data(3,pck.idx2)) + STR$(plane.data(4,pck.idx2))
    IF plane.data(0,pck.idx) = 2 THEN tt1$ = " un identified aircraft "
    IF plane.data(0,pck.idx2) = 2 THEN tt2$ = " un identified aircraft "

    IF params(3) <> 1 THEN ndx = pck.idx: GOSUB inverse.video
    IF params(3) <> 1 THEN ndx = pck.idx2: GOSUB inverse.video
    tts$ = tt1$ + " and " + tt2$ +" HAVE CRASHED -- CALL NTSB AT ONCE"
    voice.type = -1
    GOSUB speakit
    IF params(3) <> 1 THEN ndx = pck.idx2: GOSUB inverse.video
    IF params(3) <> 1 THEN ndx = pck.idx: GOSUB inverse.video
    k = pck.idx
    GOSUB paint.it.white
    k = pck.idx2
    GOSUB paint.it.white
    FOR pcr = 0 TO plane.params: plane.data(pcr,pck.idx) = 0:
plane.data(pcr,pck.idx2) = 0: NEXT    :'make inactive
    planes.crashed.total = planes.crashed.total + 2  :'tally
    warn.plane(pck.idx) = 1
    warn.plane(pck.idx2) = 1

    proc.continue:
    proxck.loop.exit:
  NEXT
  proxck1.loop.exit:
NEXT
RETURN

REM  ***********************************************
read.params:   'read the simulation parameters

'  1 = demo mode (1=demo, 0=normal)
'  2 = auto-turn mode (1=yes, 0=no)
'  3 = computer failure (1=yes, 0=no)
'  4 = speak (1=yes, 0=no)
'  5 = skill level (1-5, 1=hard, 5=easy)
'  6 = position type (1=mix, 2=enroute, 3=approach)
'  7 = controller voice level (1=low, 2=med, 3=high)
'  8 = voice speed (1-5, 1=slow, 5=fast)
'  9 = planes to start with
' 10 = random flag (1=random, 0=fixed)
' 11 = planes per shift
' 12 = random flag (1=random, 0=fixed)
' 13 = number of seconds between updates
' 14 = conflict advisory (1=yes, 0=no)

param.file$ = FILES$(1,"TEXT")
IF param.file$ = "" THEN STOP
OPEN param.file$ FOR INPUT AS #1

INPUT #1,random.seed
IF random.seed = 0 THEN RANDOMIZE TIMER ELSE RANDOMIZE random.seed
INPUT #1,mac.screen.width,mac.screen.height
IF mac.screen.width = 0 THEN mac.screen.width = SYSTEM(5)
IF mac.screen.height = 0 THEN mac.screen.height = SYSTEM(6)
INPUT #1,timelimit
INPUT #1,center.name$
INPUT #1,airport$(1)
INPUT #1,airport$(3)
airport$(2)=airport$(1): airport$(4)=airport$(3)
FOR i = 0 TO 12
  INPUT #1,carrier.size(i),carrier$(i),carrier.names$(i)
  carrier$(i)=LEFT$(carrier$(i),2)
NEXT
INPUT #1,params(1),params(2),params(3),params(4),params(14)
INPUT #1,params(5),params(6),params(7),params(8)
INPUT #1,params(9),params(10),params(11),params(12),params(13)
INPUT #1,controller.freq(1),controller.freq(2),controller.freq(3)
INPUT
#1,voice.speed(1),voice.speed(2),voice.speed(3),voice.speed(4),voice.speed(5)
INPUT #1,max.planes
INPUT #1,ap1x,ap1y,ap2x,ap2y
INPUT #1,aw3618x1,aw0927y1,aw0422y1,aw1331x1
aw3618y1 = 0: aw3618x2 = aw3618x1: aw3618y2 = mac.screen.height - 26
aw0927x1 = 0: aw0927x2 = mac.screen.width - 4: aw0927y2 = aw0927y1
aw0422x1 = 0: aw0422x2 = aw0422y1: aw0422y2 = 0
aw1331y1 = 0: aw1331x2 = mac.screen.height-26+aw1331x1: aw1331y2 =
mac.screen.height-26
IF aw0422y1 <= mac.screen.width THEN aw4ep$ = "N" ELSE aw4ep$ = "E"
IF aw1331x2 <= mac.screen.width THEN aw13ep$ = "S" ELSE aw13ep$ = "E"
INPUT #1,adv.cycles
INPUT #1,handoff.inset,handoff.len,handoff.dashsx, handoff.dashsy
INPUT #1,cen$(1),freq$(1)
INPUT #1,cen$(2),freq$(2)
INPUT #1,cen$(3),freq$(3)
INPUT #1,cen$(4),freq$(4)
INPUT #1,dswt.fdb,dswt.a,dswt.r,dswt.v,dswt.cz,dswt.hz,dswt.g,dswt.rb,dswt.hb
INPUT #1,num.glines
IF num.glines = 0 GOTO readp.exit1
FOR i = 1 TO num.glines
  LINE INPUT #1,irec$
  IF LEFT$(irec$,1) = CHR$(34) THEN irec$=RIGHT$(irec$,LEN(irec$)-1)
  IF RIGHT$(irec$,1) = CHR$(34) THEN irec$=LEFT$(irec$,LEN(irec$)-1)
  save.irec$(i) = irec$
  a = INSTR(irec$,",")
  num.pts(i) = VAL(LEFT$(irec$,a-1))
  start.pt = a + 1
  FOR j = 1 TO num.pts(i)
    a = INSTR(start.pt,irec$,",")
    geo.points(1,j,i) = VAL(MID$(irec$,start.pt,a-start.pt))
    start.pt = a + 1
    a = INSTR(start.pt,irec$,",")
    IF a = 0 THEN a = LEN(irec$)+1
    geo.points(2,j,i) = VAL(MID$(irec$,start.pt,a-start.pt))
    start.pt = a + 1
  NEXT
NEXT

readp.exit1:

INPUT #1,num.geo.text
IF num.geo.text = 0 GOTO readp.exit2
FOR i = 1 TO num.geo.text
  INPUT #1,geo.text.loc(1,i),geo.text.loc(2,i),geo.text$(i)
NEXT

readp.exit2:

CLOSE #1

RETURN

REM  **********************************************
redraw.one.plane:    'redraw one plane on the screen after a command

'mouseplane is the index for the plane to redraw

'save current new values/put in original values
FOR t = 0 TO plane.params:
new.plane.data(t)=plane.data(t,mouseplane):plane.data(t,mouseplane)=old.plane.da
ta(t):NEXT

'erase the block around the plane
x = old.plane.data(1): y = old.plane.data(2): z = old.plane.data(9): zz =
old.plane.data(20)
CALL TEXTMODE (3):  gr.color = 30
LINE (x+id.block.data(5,z,zz)-2,y+id.block.data(6,z,zz)-9) -
(x+id.block.data(5,z,zz)+36,y+id.block.data(6,z,zz)+10),gr.color,B

'erase plane from screen
k=mouseplane: GOSUB paint.it.white

'restore new values
FOR t = 0 TO plane.params: plane.data(t,mouseplane)=new.plane.data(t):NEXT

'repaint the plane with new values
k=mouseplane: GOSUB paint.it.black

'repaint any planes that may have been overwritten by the command window
FOR t = 0 TO max.index
```

```
  IF plane.data(0,t) <> 0 AND plane.data(2,t) < 40 THEN k=t: GOSUB
paint.it.black
NEXT

RETURN

REM  ***********************************************
release.from.hold:   'see if it is time to stop holding

FOR rfh = 0 TO max.index
  IF plane.data(0,rfh) <> 1 GOTO rfh.exit    :'inactive or unidentified, ignore
it
  IF plane.data(17,rfh) < 1 GOTO rfh.exit    :'only care about holding/cycles
  IF plane.data(17,rfh) > 1 THEN plane.data(17,rfh) = plane.data(17,rfh) - 1:
GOTO rfh.exit
  sayn! = plane.data(7,rfh)
  GOSUB saynum
  tts$ = carrier.names$(plane.data(3,rfh))+STR$(plane.data(4,rfh))+" IS
PROCEEDING ON HEADING "+sayn$+" DEGREES "
  voice.type = rfh
  GOSUB speakit
  k = rfh
  GOSUB paint.it.white
  plane.data(17,rfh) = 0
  k = rfh
  GOSUB paint.it.black

  rfh.exit:
NEXT
RETURN

REM  ***********************************************
save.options:   'writes the current options to the selected file

IF INSTR(param.file$,":") > 0 THEN
  FOR isr = LEN(param.file$) TO 1 STEP -1
    IF MID$(param.file$,isr,1) = ":" GOTO found.colon
  NEXT
  found.colon: param.file$=RIGHT$(param.file$,LEN(param.file$)-isr)
END IF

save.file$ = FILES$(0,param.file$)
IF save.file$ = "" THEN RETURN ELSE param.file$ = save.file$

OPEN param.file$ FOR OUTPUT AS #1

WRITE #1,random.seed
WRITE #1,mac.screen.width,mac.screen.height
WRITE #1,timelimit
WRITE #1,center.name$
WRITE #1,airport$(1)
WRITE #1,airport$(3)
FOR i = 0 TO 12
  WRITE #1,carrier.size(i),carrier$(i),carrier.names$(i)
NEXT
```

```
WRITE #1,params(1),params(2),params(3),params(4),params(14)
WRITE #1,params(5),params(6),params(7),params(8)
WRITE #1,params(9),params(10),params(11),params(12),params(13)
WRITE #1,controller.freq(1),controller.freq(2),controller.freq(3)
WRITE
#1,voice.speed(1),voice.speed(2),voice.speed(3),voice.speed(4),voice.speed(5)
WRITE #1,max.planes
WRITE #1,ap1x,ap1y,ap2x,ap2y
WRITE #1,aw3618x1,aw0927y1,aw0422y1,aw1331x1
WRITE #1,adv.cycles
WRITE #1,handoff.inset,handoff.len,handoff.dashsx,handoff.dashsy
WRITE #1,cen$(1),freq$(1)
WRITE #1,cen$(2),freq$(2)
WRITE #1,cen$(3),freq$(3)
WRITE #1,cen$(4),freq$(4)
WRITE #1,dswt.fdb,dswt.a,dswt.r,dswt.v,dswt.cz,dswt.hz,dswt.g,dswt.rb,dswt.hb
WRITE #1,num.glines
IF num.glines = 0 GOTO savep.exit1
FOR i = 1 TO num.glines
  WRITE #1,save.irec$(i)
NEXT

savep.exit1:
WRITE #1,num.geo.text
IF num.geo.text = 0 GOTO savep.exit2
FOR i = 1 TO num.geo.text
  WRITE #1,geo.text.loc(1,i),geo.text.loc(2,i),geo.text$(i)
NEXT

savep.exit2:
CLOSE #1

RETURN

REM  ***********************************************
saynum:   'convert a number (sayn!) into speech (sayn$)

sayn$=""
IF params(4) = 0 THEN RETURN  :'voice is off
IF sayn!=0 THEN sayn$=" zeero ": RETURN
IF sayn! < 1000 GOTO saynum.a
sayn1 = INT(sayn!/1000)
IF sayn1 < 20 THEN sayn$=units$(sayn1)+" thousend ": GOTO saynum.a
sayn2 = INT(sayn1/10): sayn3 = sayn1-(10*sayn2)
sayn$ = sayn$ + tens$(sayn2)
IF sayn3 > 0 THEN sayn$ = sayn$ + units$(sayn3)
sayn$ = sayn$ + " thousend "
saynum.a:
sayn! = sayn! - (INT(sayn!/1000)*1000)
IF sayn! >= 100 THEN sayn$ = sayn$ + units$(INT(sayn!/100)) + " hun dred "
sayn! = sayn! - (INT(sayn!/100)*100)
IF sayn! = 0 THEN RETURN
IF sayn! < 20 THEN sayn$ = sayn$ + units$(sayn!): RETURN
sayn2 = INT(sayn!/10): sayn3 = sayn! - (10*sayn2)
sayn$ = sayn$ + tens$(sayn2)
```

```
IF sayn3 > 0 THEN sayn$ = sayn$ + units$(sayn3)
RETURN

REM  **********************************************
selective.display:    'handle user-selectable displays on radar screen

ON menu.item GOTO sd.fdb,sd.a,sd.r,sd.v,sd.cz,sd.hz,sd.g,sd.rb,sd.hb
GOTO sd.exit

sd.fdb:
IF dswt.fdb = 0 THEN dswt.fdb = 1 ELSE dswt.fdb = 0
MENU 3,1,dswt.fdb+1,"full data blocks"
GOTO sd.exit

sd.a:
IF dswt.a = 0 THEN dswt.a = 1 ELSE dswt.a = 0
MENU 3,2,dswt.a+1,"airways"
GOTO sd.exit

sd.r:
IF dswt.r = 0 THEN dswt.r = 1 ELSE dswt.r = 0
MENU 3,3,dswt.r+1,"runways"
GOTO sd.exit

sd.v:
IF dswt.v = 0 THEN dswt.v = 1 ELSE dswt.v = 0
MENU 3,4,dswt.v+1,"VORs"
GOTO sd.exit

sd.cz:
IF dswt.cz = 0 THEN dswt.cz = 1 ELSE dswt.cz = 0
MENU 3,5,dswt.cz+1,"control zones"
GOTO sd.exit

sd.hz:
IF dswt.hz = 0 THEN dswt.hz = 1 ELSE dswt.hz = 0
MENU 3,6,dswt.hz+1,"handoff zones"
GOTO sd.exit

sd.g:
IF dswt.g = 0 THEN dswt.g = 1 ELSE dswt.g = 0
MENU 3,7,dswt.g+1,"geography"
GOTO sd.exit

sd.rb:
IF dswt.rb = 0 THEN dswt.rb = 1 ELSE dswt.rb = 0
MENU 3,8,dswt.rb+1,"runway barricades"
GOTO sd.exit

sd.hb:
IF dswt.hb = 0 THEN dswt.hb = 1 ELSE dswt.hb = 0
MENU 3,9,dswt.hb+1,"handoff barricades"
GOTO sd.exit

sd.exit:
```

```
GOSUB paint.with.clear

RETURN

REM  **********************************************
set.vor.hdg:   'if the plane is on a heading to a VOR, set current heading

FOR vck = 0 TO max.index
  IF plane.data(0,vck) <> 1 GOTO vck.exitt    :'inactive or unidentified, ignore
it
  IF plane.data(12,vck) >= 0 GOTO vck.exitt    :'not on vor heading
  oldh = plane.data(7,vck)
  newh = oldh
  vcp = ABS(plane.data(12,vck))    :'-1,-2,-3,-4
  REM see if plane is already at the VOR
  IF plane.data(1,vck) < vor.ck.params(1,vcp) - 3 OR plane.data(1,vck) >
vor.ck.params(1,vcp) + 3 GOTO vck.not.at.vor.yet
  IF plane.data(2,vck) < vor.ck.params(2,vcp) - 3 OR plane.data(2,vck) >
vor.ck.params(2,vcp) + 3 GOTO vck.not.at.vor.yet
  REM the plane is at the VOR, set hold flag
  IF plane.data(17,vck) <> 0 GOTO vck.exit
  tts$ = carrier.names$(plane.data(3,vck))+STR$(plane.data(4,vck))+"  IS HOLDING
AT "+vor.names$(vcp)
  voice.type = vck
  GOSUB speakit
  k = vck
  GOSUB paint.it.white
  plane.data(17,vck) = -1
  plane.data(12,vck) = jetway.check.data(ABS(plane.data(15,vck)))  :'final
heading
  k = vck
  GOSUB paint.it.black
  GOTO vck.exit
  vck.not.at.vor.yet:
  REM set current heading based on vor position and current position
  IF plane.data(1,vck) >= vor.ck.params(1,vcp) - 3 AND plane.data(1,vck) <=
vor.ck.params(1,vcp) + 3 GOTO vck.inx.area
  IF plane.data(2,vck) >= vor.ck.params(2,vcp) - 3 AND plane.data(2,vck) <=
vor.ck.params(2,vcp) + 3 GOTO vck.iny.area
  IF plane.data(1,vck) < vor.ck.params(1,vcp) AND plane.data(2,vck) <
vor.ck.params(2,vcp) GOTO vck.quad1
  IF plane.data(1,vck) > vor.ck.params(1,vcp) AND plane.data(2,vck) <
vor.ck.params(2,vcp) GOTO vck.quad2
  IF plane.data(1,vck) < vor.ck.params(1,vcp) AND plane.data(2,vck) >
vor.ck.params(2,vcp) GOTO vck.quad3
  IF plane.data(1,vck) > vor.ck.params(1,vcp) AND plane.data(2,vck) >
vor.ck.params(2,vcp) GOTO vck.quad4
  vck.quad1:
  newh = 135
  GOTO vck.exit
  vck.quad2:
  newh = 225
  GOTO vck.exit
  vck.quad3:
  newh = 45
  GOTO vck.exit
  vck.quad4:
  newh = 315
  GOTO vck.exit
  vck.inx.area:
  IF plane.data(2,vck) > vor.ck.params(2,vcp) THEN newh= 0
  IF plane.data(2,vck) < vor.ck.params(2,vcp) THEN newh= 180
  GOTO vck.exit
  vck.iny.area:
  IF plane.data(1,vck) > vor.ck.params(1,vcp)  THEN newh = 270
  IF plane.data(1,vck) < vor.ck.params(1,vcp)  THEN newh = 90

  vck.exit:
  IF newh = oldh GOTO vck.exitt
  sayn! = newh
  GOSUB saynum
  tts$ = carrier.names$(plane.data(3,vck))+STR$(plane.data(4,vck))+"  IS TURNING
TO HEADING "+sayn$+" DEGREES "
  voice.type = vck
  GOSUB speakit
  k = vck
  GOSUB paint.it.white
  plane.data(7,vck) = newh
  k = vck
  GOSUB paint.it.black

  vck.exitt:
NEXT
RETURN

REM  **********************************************
set.cfa.hdg:   'if the plane is cleared for approach, check for arrival at VOR

FOR vck = 0 TO max.index
  IF plane.data(0,vck) <> 1 GOTO cfa.exit    :'inactive or unidentified, ignore
it
  IF plane.data(22,vck) <> 1 GOTO cfa.exit    :'not cleared for approach
  'if at the VOR and not holding and not on final hdg, set final heading and
hold flag
  vcp = ABS(plane.data(15,vck)+8)    :'-1,-2,-3,-4
  IF plane.data(1,vck) < vor.ck.params(1,vcp) - 3 OR plane.data(1,vck) >
vor.ck.params(1,vcp) + 3 GOTO cfa.exit
  IF plane.data(2,vck) < vor.ck.params(2,vcp) - 3 OR plane.data(2,vck) >
vor.ck.params(2,vcp) + 3 GOTO cfa.exit
  'at the VOR
  k = vck: GOSUB paint.it.white
  plane.data(10,vck) = 10
  plane.data(11,vck) = 10
  IF plane.data(12,vck) <>jetway.check.data(ABS(plane.data(15,vck))) THEN
plane.data(17,vck)=-1
  plane.data(12,vck) =jetway.check.data(ABS(plane.data(15,vck)))
  k = vck: GOSUB paint.it.black
  IF plane.data(5,vck) > 25 OR plane.data(6,vck) > 25 THEN plane.data(17,vck)=-
1:GOTO cfa.exit
  IF plane.data(17,vck)=0 GOTO cfa.exit
  plane.data(17,vck) = 0
  tts$ = carrier.names$(plane.data(3,vck))+STR$(plane.data(4,vck))+"  IS ON
FYENUL FOR RUNWAY " +  jetway.check.name$(ABS(plane.data(15,vck)))
  voice.type = vck
  GOSUB speakit

  cfa.exit:
NEXT
RETURN

REM  ***********************************************
speakit:   'pronounce the string in tts$

IF params(4) = 0 THEN RETURN  :'don't say anything
IF voice.type = -1 THEN srate% = controller.freq(params(7)) ELSE srate% =
plane.data(19,voice.type)
speechpitch! speechhand!,srate%,0
readerh! speechhand!,tts$,phonh!,speecherr%
soundouth! speechhand!,phonh!,speecherr%
disposehandle! phonh!
RETURN

REM  ***********************************************
speak.text:   'test pronounces center name, airports, or airlines

IF button.num = 1 THEN tts$ = center.name$+" center"
IF button.num = 2 THEN tts$ = "welcome to "+airport$(1)
IF button.num = 3 THEN tts$ = "welcome to "+airport$(3)
IF button.num > 3 THEN tts$ = carrier.names$(button.num-17)
voice.type = -1: GOSUB speakit

RETURN

REM  ***********************************************
speak.invalid.msg:   'bad user input

tts$ = "invalid entry"
IF ecode = 0 GOTO sim.exit

ON ecode GOTO alt.ermsg, spd.ermsg, hdg.ermsg, path.ermsg, wrn.ermsg, db.ermsg,
ldr.ermsg, hold.ermsg, vor.ermsg, both.ermsg, divert.ermsg, clr.ermsg, ho.ermsg,
kp.ermsg
GOTO sim.exit

alt.ermsg:
tts$ = "al titude must be between "
IF carrier.size(plane.data(3,mouseplane)) = 1 THEN tts$=tts$+"40 and 400" ELSE
tts$=tts$+"15 and 150"
GOTO sim.exit

spd.ermsg:
tts$ = "speed must be between "
IF carrier.size(plane.data(3,mouseplane)) = 1 THEN tts$=tts$+"20 and 70" ELSE
tts$=tts$+"10 and 20"
GOTO sim.exit
```

```
hdg.ermsg:
tts$ = "heading must be between 0 and 359"
GOTO sim.exit

path.ermsg:
tts$ = "you can only project the path for a maximum of fifteen menits"
GOTO sim.exit

wrn.ermsg:
tts$ = "you can only spessafy a warning circle for a maximum of fifty syekils"
GOTO sim.exit

db.ermsg:
tts$ = "5 is an invalid entry, use 1 through 4 and 6 through 9 only"
GOTO sim.exit

ldr.ermsg:
tts$ = "the leader line cannot be in creesed or de creesed by that amount"
GOTO sim.exit

hold.ermsg:
tts$ = "you can only spessafy a hold for a maximum of fifty syekils"
GOTO sim.exit

vor.ermsg:
tts$ = "invalid VORE number, use only eighteen, thirty six, nine, or twenty
seven"
GOTO sim.exit

both.ermsg:
tts$ = "that val-you is outside of the al titude or speed range for this type of
aircraft"
GOTO sim.exit

divert.ermsg:
tts$ = "you cannot divert this plane to that runway"
GOTO sim.exit

clr.ermsg:
tts$ = "you cannot clear this plane for an approach since he is not landing"
GOTO sim.exit

ho.ermsg:
tts$ = "that is an im prop er handoff action for this plane"
GOTO sim.exit

kp.ermsg:
tts$ = "use keypad numbers for heading direction, 1 through 9"
GOTO sim.exit

sim.exit:
tts$=tts$+", try again"
voice.type = -1: GOSUB speakit
```

```
RETURN

REM  ***********************************************
specl.requests:   'generate special requests every so often

IF INT(RND*(2*params(5))) + 1 <> 1 THEN RETURN
IF INT(300*RND) = 153 THEN GOSUB gen.comp.failure: GOTO srx
srid = INT(100*RND) + 1  :'20% old plane, 1% mistake, 79% new plane
IF srid <= 20 THEN GOSUB specl.req.oldplane :GOTO srx
IF srid = 39 THEN GOSUB specl.req.mistake :GOTO srx
GOSUB specl.req.newplane
srx:
RETURN

REM  ***********************************************
specl.req.mistake:   'generate a mistake for a plane

REM generate a mistake for an existing plane
REM generate a change in assigned altitude or speed
REM don't do a mistake for a plane close to the edge of the screen, though
REM don't do a mistake for a plane on a runway heading
REM don't do a mistake for an unidentified plane

num.old.planes = 0
FOR sr = 0 TO max.index
  IF plane.data(0,sr) = 1 THEN num.old.planes = num.old.planes + 1
NEXT
IF num.old.planes = 0 THEN GOTO sr.mistake.exit    :'there are no active old
planes, so don't do anything

REM determine which plane to generate a mistake for
which.one = INT(num.old.planes * RND) + 1
REM now find that one
pcount = 0
FOR sr.idx = 0 TO max.index
  IF plane.data(0,sr.idx) <> 1 GOTO sr.check.next.mistake
  pcount = pcount + 1
  IF pcount = which.one GOTO chk.mistake.gotit
  sr.check.next.mistake:
NEXT
GOTO sr.mistake.exit  :'error, couldn't find plane which.one
chk.mistake.gotit:
REM sr.idx now contains the index of the plane to do a mistake on
REM if close to the edge, don't do anything
IF plane.data(1,sr.idx) < 15 OR plane.data(1,sr.idx) > mac.screen.width-19 GOTO
sr.mistake.exit
IF plane.data(2,sr.idx) < 15 OR plane.data(2,sr.idx) > mac.screen.height-41 GOTO
sr.mistake.exit
REM if on runway heading, don't do anything
IF plane.data(15,sr.idx) <= -9 GOTO sr.mistake.exit
REM determine the type of request (1=alt, 2=spd)
sreq.type = INT(2*RND)+1
IF sreq.type = 1 GOTO gen.alt.change.mistake
IF sreq.type = 2 GOTO gen.spd.change.mistake
GOTO sr.mistake.exit  :'error
```

```
gen.alt.change.mistake:
IF carrier.size(plane.data(3,sr.idx)) = 1 THEN req.alt = INT(360*RND)+40
IF carrier.size(plane.data(3,sr.idx)) = 0 THEN req.alt = INT(135*RND)+15
k = sr.idx
GOSUB paint.it.white
plane.data(10,sr.idx) = req.alt
k = sr.idx
GOSUB paint.it.black
GOTO sr.mistake.exit

gen.spd.change.mistake:
IF carrier.size(plane.data(3,sr.idx)) = 1 THEN req.spd = INT(50*RND)+20
IF carrier.size(plane.data(3,sr.idx)) = 0 THEN req.spd = INT(10*RND)+10
k = sr.idx
GOSUB paint.it.white
plane.data(11,sr.idx) = req.spd
k = sr.idx
GOSUB paint.it.black
GOTO sr.mistake.exit

sr.mistake.exit:
RETURN

REM  ***********************************************
specl.req.newplane:   'generate all parameters for a new plane

IF planes.activated >= params(11) GOTO sr.new.exit  :'shift over
FOR sr = 0 TO max.planes-1  :'must search entire array for avail space
  IF plane.data(0,sr) = 0 GOTO sr.new.continue
NEXT
GOTO sr.new.exit    :'there is no room for new planes, so don't do anything
sr.new.continue:

REM index sr points to the position to use for the new plane in array plane.data
j = sr
GOSUB make.new.plane
IF INT(10*RND)+1 = 5 THEN plane.data(0,sr) = 2: planes.activated =
planes.activated - 1: GOTO new.unident
greeting = INT(3*RND)
IF greeting = 0 THEN greeting.msg$ = "hello "+center.name$+" center, ": GOTO
greeting.continue
IF greeting = 1 THEN greeting.msg$ = center.name$+" center, ": GOTO
greeting.continue
greeting.hour = VAL(LEFT$(TIME$,2))
IF greeting.hour < 12 THEN greeting.msg$ = "good morning "+center.name$+"
center, "
IF greeting.hour >= 12 AND greeting.hour <= 17 THEN greeting.msg$ = "good
afternoon "+center.name$+" center, "
IF greeting.hour > 17 THEN greeting.msg$ = "good eevning "+center.name$+"
center, "

greeting.continue:
tts$ = greeting.msg$
```

```
tts$ = tts$ + carrier.names$(plane.data(3,sr))+STR$(plane.data(4,sr))+" is with
you "
sayn! = plane.data(15,sr)
GOSUB saynum
IF plane.data(15,sr) >= 0 THEN tts$ = tts$ + "on a heading of "+sayn$+ " DEGREES
": GOTO newp.cont
IF plane.data(15,sr) >= -8 THEN tts$ = tts$ + " requesting vectors to
"+jetway.check.name$(ABS(plane.data(15,sr))): GOTO newp.cont
tts$ = tts$ + "requesting vectors to land at "+airport$(ABS(plane.data(15,sr))-
8)+" ON RUNWAY "+jetway.check.name$(ABS(plane.data(15,sr)))
newp.cont:
k = sr
GOSUB paint.it.black
ndx = sr: GOSUB inverse.video
voice.type = sr
GOSUB speakit
ndx = sr: GOSUB inverse.video
GOTO sr.new.exit
new.unident:
k = sr
GOSUB paint.it.black
sr.new.exit:
RETURN

REM  **********************************************
specl.req.oldplane:   'generate a change in alt/spd/hdg for an old plane

num.old.planes = 0
FOR sr = 0 TO max.index
  IF plane.data(0,sr) = 1 THEN num.old.planes = num.old.planes + 1
NEXT
IF num.old.planes = 0 THEN GOTO sr.old.exit    :'there are no active old planes,
so don't do anything

REM determine which plane wants to issue a special request
which.one = INT(num.old.planes * RND) + 1
REM now find that one
pcount = 0
FOR sr.idx = 0 TO max.index
  IF plane.data(0,sr.idx) <> 1 GOTO sr.check.next.old
  pcount = pcount + 1
  IF pcount = which.one GOTO chk.old.gotit
  sr.check.next.old:
NEXT
GOTO sr.old.exit  :'error, couldn't find plane which.one
chk.old.gotit:
REM sr.idx now contains the index of the plane making the special request
voice.type = sr.idx  :'for subsequent speakit call
REM don't change planes on headings for runways
IF plane.data(15,sr.idx) <= -9 GOTO sr.old.exit
REM don't change planes that are handed off
IF plane.data(23,sr.idx) = 2 GOTO sr.old.exit
REM determine the type of request (1=alt, 2=spd, 3=hdg)
sreq.type = INT(3*RND)+1
IF sreq.type = 1 GOTO gen.alt.change
IF sreq.type = 2 GOTO gen.spd.change
IF sreq.type = 3 GOTO gen.hdg.change
GOTO sr.old.exit  :'error

gen.alt.change:
IF carrier.size(plane.data(3,sr.idx)) = 1 THEN req.alt = INT(360*RND)+40
IF carrier.size(plane.data(3,sr.idx)) = 0 THEN req.alt = INT(135*RND)+15
sayn! = req.alt*100!
GOSUB saynum
tts$ = carrier.names$(plane.data(3,sr.idx))+STR$(plane.data(4,sr.idx))
tts$ = tts$ + " REQUESTS AL TITUDE OF"+sayn$+" FEET"
GOSUB speakit
k = sr.idx
GOSUB paint.it.white
plane.data(13,sr.idx) = req.alt
k = sr.idx
GOSUB paint.it.black
GOTO sr.wait.for.click

gen.spd.change:
IF carrier.size(plane.data(3,sr.idx)) = 1 THEN req.spd = INT(50*RND)+20
IF carrier.size(plane.data(3,sr.idx)) = 0 THEN req.spd = INT(10*RND)+10
sayn! = req.spd*10
GOSUB saynum
tts$ = carrier.names$(plane.data(3,sr.idx))+STR$(plane.data(4,sr.idx))
tts$ = tts$ + " REQUESTS SPEED OF"+sayn$+" KNOTS"
GOSUB speakit
k = sr.idx
GOSUB paint.it.white
plane.data(14,sr.idx) = req.spd
k = sr.idx
GOSUB paint.it.black
GOTO sr.wait.for.click

gen.hdg.change:
ii = INT(2*RND)  :'req. heading = 50% a heading, 50% a jetway/runway
IF ii > 0 THEN GOTO sr.new.degrees
IF params(6) = 1 THEN req.hdg = -(INT(12*RND)+1)  :'jetways/runways are negative
numbers
IF params(6) = 2 THEN req.hdg = -(INT(8*RND)+1)   :'jetways only
IF params(6) = 3 THEN req.hdg = -(INT(4*RND)+1+8) :'runways only
IF params(1) = 1 AND req.hdg = -5 THEN req.hdg = -1
IF params(1) = 1 AND req.hdg = -6 THEN req.hdg = -2
IF params(1) = 1 AND req.hdg = -7 THEN req.hdg = -3
IF params(1) = 1 AND req.hdg = -8 THEN req.hdg = -4
IF plane.data(5,sr.idx) > 60 AND req.hdg <= -9 THEN GOTO sr.old.exit  :'no
runway if too high
tts$ = carrier.names$(plane.data(3,sr.idx))+STR$(plane.data(4,sr.idx))
IF req.hdg >= -8 THEN tts$ = tts$ + " REQUESTS VECTORS TO
"+jetway.check.name$(ABS(req.hdg))
IF req.hdg <  -8 THEN tts$ = tts$ + " requests vectors to land at
"+airport$(ABS(req.hdg)-8)+" ON RUNWAY "+jetway.check.name$(ABS(req.hdg))
GOSUB speakit
k = sr.idx
GOSUB paint.it.white
```

```
plane.data(15,sr.idx) = req.hdg
IF req.hdg <= -9 THEN plane.data(13,sr.idx) = 10: plane.data(14,sr.idx) = 10
:'asgd alt/spd
k = sr.idx
GOSUB paint.it.black
GOTO sr.wait.for.click

sr.new.degrees:
req.hdg = INT(359*RND)
sayn! = req.hdg
GOSUB saynum
tts$ = carrier.names$(plane.data(3,sr.idx))+STR$(plane.data(4,sr.idx))
tts$ = tts$ + " REQUESTS HEADING OF"+sayn$+" DEGREES "
GOSUB speakit
k = sr.idx
GOSUB paint.it.white
plane.data(15,sr.idx) = req.hdg
k = sr.idx
GOSUB paint.it.black

sr.wait.for.click:
sr.old.exit:
RETURN

REM  **********************************************
startup:   'display the startup screen

WINDOW 1,,(2,22)-(510,338),-3
CALL TEXTFACE (1)
LOCATE 3,11
DrawText! "Professional"
LOCATE 4,4
DrawText! "Air Traffic Controller Simulator"
LOCATE 6,5
CALL TEXTFACE (0)
DrawText! "           Version 1.2"
LOCATE 7,5
DrawText! "       Copyright © 1988"
LOCATE 9,5
DrawText! " Advanced Simulation Systems"
LOCATE 10,5
DrawText! " Post Office Box 756"
LOCATE 11,5
DrawText! " Huntingtown, MD 20639"
LOCATE 13,5
DrawText! " portions of this program are"
LOCATE 14,5
DrawText! " © 1985 Apple Computer, Inc."
LOCATE 15,5
DrawText! " © 1985 Clear Lake Research, Inc."
LOCATE 17,5
DrawText! " Program written by Don Shepherd"

BUTTON 1,1,"BEGIN",(275,280)-(375,310),1
BUTTON 2,1,"DEMO MODE",(400,280)-(500,310),1
BUTTON 3,1,"SET OPTIONS",(275,240)-(375,270),1
IF params(4) = 1 THEN BUTTON 4,1,"SHUT UP",(400,240)-(500,270),1
IF params(4) = 0 THEN BUTTON 4,1,"TALK TO ME",(400,240)-(500,270),1

tts$ = "welcome to professional air traffic controller simulaytor for the
"+center.name$+" center"
voice.type = -1  :'-1 causes normal voice
GOSUB speakit

ON DIALOG GOSUB startup.dialog
DIALOG ON

xc = 146: yc = -10
FOR i = 0 TO 75 STEP 25
  oval(0)=30+i+yc
  oval(1)=140+i+xc
  oval(2)=230-i+yc
  oval(3)=340-i+xc
  CALL FRAMEOVAL(VARPTR(oval(0)))
NEXT
LINE (386,20)-(386,219)
LINE (286,120)-(485,120)
oval(0)=31+yc
oval(1)=141+xc
oval(2)=229+yc
oval(3)=339+xc
FOR i = 1 TO 10
  x=165+xc+INT(150*RND)
  y=55+yc+INT(150*RND)
  PSET(x,y)
  PSET(x+1,y)
  PSET(x,y+1)
  PSET(x+1,y+i)
NEXT
CALL INVERTARC (VARPTR(oval(0)),0,5)

xwait = 1
msgnum = 0: voice.type = -1

startup.loop:
FOR ang = 0 TO 355 STEP 5
  CALL INVERTARC(VARPTR(oval(0)),ang,5)
  CALL INVERTARC(VARPTR(oval(0)),ang+5,5)
  IF xwait = 0 GOTO startup.exit
NEXT
tts$ = demo.messages$(msgnum)
GOSUB speakit
msgnum = msgnum + 1: IF msgnum = 10 THEN msgnum = 0
IF xwait = 1 GOTO startup.loop

startup.exit:

WINDOW CLOSE 1
DIALOG OFF
```

```
RETURN

REM  ***********************************************
startup.dialog:   'handle initial startup screen

event.type = DIALOG(0)
button.num = DIALOG(1)
IF button.num = 2 THEN params(1) = 1: MENU 4,0,1,"Demonstration Mode"
IF button.num = 3 THEN options = 1
IF button.num = 4 AND params(4) = 1 THEN params(4) = 0: BUTTON 4,1,"TALK TO
ME",(400,240)-(500,270),1: RETURN
IF button.num = 4 AND params(4) = 0 THEN params(4) = 1: BUTTON 4,1,"SHUT
UP",(400,240)-(500,270),1: RETURN
xwait = 0

RETURN

REM  ***********************************************
stop.Nsec.delay:   'stop the delay cycle

delay.switch = 1
RETURN

REM  ***********************************************
tca.violation.check:    'see if a plane violates the control zone of an airport

IF debug.switch = -1 THEN RETURN

FOR tca = 0 TO max.index
  IF plane.data(0,tca) <> 1 GOTO tca.exit    :'inactive or unidentified, ignore
it
  IF plane.data(5,tca) >= 50 GOTO tca.exit    :'above 5000 feet, so ignore
  IF (plane.data(1,tca)>ap2x-61 AND plane.data(1,tca)<ap2x+91) AND
(plane.data(2,tca)>ap2y-40 AND plane.data(2,tca)<ap2y+42) GOTO in.zone1
  IF (plane.data(1,tca)>ap1x-40 AND plane.data(1,tca)<ap1x+42) AND
(plane.data(2,tca)>ap1y-61 AND plane.data(2,tca)<ap1y+91) GOTO in.zone2
  GOTO tca.exit    :'not in either control zone
  in.zone1:
  IF plane.data(15,tca) = -11 OR plane.data(15,tca) = -12 GOTO tca.exit
:'supposed to be there
  IF (plane.data(15,tca) = -9 OR plane.data(15,tca) = -10) AND
(plane.data(1,tca)>ap1x-40 AND plane.data(1,tca)<ap1x+42) AND
(plane.data(2,tca)>ap1y-61 AND plane.data(2,tca)<ap1y+91) GOTO tca.exit
  GOTO tca.viol
  in.zone2:
  IF plane.data(15,tca) = -9 OR plane.data(15,tca) = -10 GOTO tca.exit
:'supposed to be there
  IF (plane.data(15,tca) = -11 OR plane.data(15,tca) = -12) AND
(plane.data(1,tca)>ap2x-61 AND plane.data(1,tca)<ap2x+91) AND
(plane.data(2,tca)>ap2y-40 AND plane.data(2,tca)<ap2y+42) GOTO tca.exit
  tca.viol:
  planes.tca.violations = planes.tca.violations + 1
  tts$ = carrier.names$(plane.data(3,tca))+STR$(plane.data(4,tca))+" HAS
VYEOLATED THE CONTROL ZONE"
  IF params(3) <> 1 THEN ndx = tca: GOSUB inverse.video
  voice.type = tca
  GOSUB speakit
  IF params(3) <> 1 THEN ndx = tca: GOSUB inverse.video
  tca.exit:
NEXT
RETURN

REM  ***********************************************
update.airline.size:   'change the airline size from dialog screen

IF carrier.size(button.num-4) = 0 THEN carrier.size(button.num-4) = 1 ELSE
carrier.size(button.num-4) = 0
n = 80 + 15*(button.num-4)
BUTTON button.num,carrier.size(button.num-4)+1,"",(2,n-1)-(20,n+12),2

RETURN

REM  ***********************************************
update.counts:   'update random switches on dialog display

IF button.num = 58 AND params(10) = 0 THEN params(10) = 1: GOTO ucx
IF button.num = 58 AND params(10) = 1 THEN params(10) = 0: GOTO ucx
IF button.num = 59 AND params(12) = 0 THEN params(12) = 1: GOTO ucx
IF button.num = 59 AND params(12) = 1 THEN params(12) = 0: GOTO ucx
ucx:
GOSUB write.modes3

RETURN

REM  ***********************************************
update.current.values:    'update current alt/speed/heading

IF plane.data(18,k) > 0 THEN plane.data(18,k) = plane.data(18,k) - 1
warn.plane(k) = 0

REM perform x-y coordinate adjustment
hdg = plane.data(7,k)
n = speedfact(plane.data(6,k))       :'x-y adjustment is based on speed and
heading
IF plane.data(17,k) <> 0 GOTO skip.pos.adj    :'holding pattern
plane.data(1,k) = plane.data(1,k) + (n*hdg.direction(4,hdg))
plane.data(2,k) = plane.data(2,k) + (n*hdg.direction(5,hdg))
skip.pos.adj:

REM perform altitude adjustment
IF plane.data(5,k) = plane.data(10,k) THEN plane.data(8,k) = 0:GOTO xy
IF plane.data(5,k) > plane.data(10,k) THEN plane.data(5,k)=plane.data(5,k)-1:
plane.data(8,k) = -1: GOTO xy
plane.data(5,k)=plane.data(5,k)+1: plane.data(8,k) = 1
xy:

REM perform speed adjustment
IF plane.data(6,k) = plane.data(11,k) GOTO xz
IF plane.data(6,k) < plane.data(11,k) THEN plane.data(6,k) = plane.data(6,k) +
1: GOTO xz
```

```
plane.data(6,k) = plane.data(6,k) - 1
xz:

REM perform heading adjustment
IF plane.data(12,k) >= 0 THEN plane.data(7,k) = plane.data(12,k)

RETURN

REM  **********************************************
update.levels:   'update parameter levels on dialog display

IF button.num = 42 THEN params(5)=1
IF button.num = 43 THEN params(5)=2
IF button.num = 44 THEN params(5)=3
IF button.num = 45 THEN params(5)=4
IF button.num = 46 THEN params(5)=5
IF button.num = 47 THEN params(8)=1:speechrate!
speechhand!,voice.speed(params(8))
IF button.num = 48 THEN params(8)=2:speechrate!
speechhand!,voice.speed(params(8))
IF button.num = 49 THEN params(8)=3:speechrate!
speechhand!,voice.speed(params(8))
IF button.num = 50 THEN params(8)=4:speechrate!
speechhand!,voice.speed(params(8))
IF button.num = 51 THEN params(8)=5:speechrate!
speechhand!,voice.speed(params(8))
IF button.num = 52 THEN params(7)=1
IF button.num = 53 THEN params(7)=2
IF button.num = 54 THEN params(7)=3
IF button.num = 55 THEN params(6)=1
IF button.num = 56 THEN params(6)=2
IF button.num = 57 THEN params(6)=3

GOSUB write.modes2

RETURN

REM  **********************************************
update.modes:   'update modes on dialog display

IF button.num = 32 THEN params(1)=1:MENU 4,0,1,"Demonstration Mode"
IF button.num = 33 THEN params(1)=0:MENU 4,0,0,""
IF button.num = 34 THEN params(2)=1
IF button.num = 35 THEN params(2)=0
IF button.num = 36 THEN params(3)=1
IF button.num = 37 THEN params(3)=0
IF button.num = 38 THEN params(4)=1
IF button.num = 39 THEN params(4)=0
IF button.num = 40 THEN params(14)=1
IF button.num = 41 THEN params(14)=0
GOSUB write.modes1

RETURN

REM  **********************************************
```

```
were.gone:    'plane has left the screen, tally and say goodbye

k = jck.idx
cur.x.pos = plane.data(1,k)
cur.y.pos = plane.data(2,k)
IF plane.data(0,k) = 2 THEN GOTO gone.exit  :'unidentified plane is gone
special.message$ = ""   :'will tell of improper alt/spd/hdg on exiting
IF plane.data(23,k) <> 2 GOTO bad.handoff.exit
IF plane.data(1,k) < 0 AND handoff.status(4) = 1 GOTO bad.handoff.exit
IF plane.data(1,k) > mac.screen.width-4 AND handoff.status(3) = 1 GOTO
bad.handoff.exit
IF plane.data(2,k) < 0 AND handoff.status(1) = 1 GOTO bad.handoff.exit
IF plane.data(2,k) > mac.screen.height-26 AND handoff.status(2) = 1 GOTO
bad.handoff.exit
IF plane.data(10,k) <> plane.data(13,k) THEN planes.exited.bad.alt =
planes.exited.bad.alt + 1: special.message$ = "-- EXITTED AT IM PROP ER AL
TITUDE --": GOTO improper.place.continue
IF plane.data(11,k) <> plane.data(14,k) THEN planes.exited.bad.spd =
planes.exited.bad.spd + 1: special.message$ = "-- EXITTED AT IM PROP ER SPEED --
": GOTO improper.place.continue
IF plane.data(15,k) < 0 GOTO exit.hdg.jetway.check
IF plane.data(12,k) <> plane.data(15,k) THEN planes.exited.bad.hdg =
planes.exited.bad.hdg + 1: special.message$ = "-- EXITTED AT IM PROP ER HEADING
--": GOTO improper.place.continue
planes.exited.successfully = planes.exited.successfully + 1  :'a good one
GOTO improper.place.continue
bad.handoff.exit:
planes.exited.bad.sector = planes.exited.bad.sector + 1
special.message$ = " -- EXXITTED WITHOUT A PROP ER HANDOFF"
GOTO improper.place.continue
exit.hdg.jetway.check:
REM check for proper jetway heading and position

IF plane.data(15,k) = -1 OR plane.data(15,k) = -2 THEN
  IF cur.x.pos >= aw3618x1-10 AND cur.x.pos <= aw3618x1+10 THEN
    IF plane.data(12,k)=jetway.check.data(ABS(plane.data(15,k))) THEN
      planes.exited.successfully = planes.exited.successfully + 1 :GOTO
improper.place.continue
    END IF
  END IF
END IF

IF plane.data(15,k) = -3 OR plane.data(15,k) = -4 THEN
  IF cur.y.pos >= aw0927y1-10 AND cur.y.pos <= aw0927y1+10 THEN
    IF plane.data(12,k)=jetway.check.data(ABS(plane.data(15,k))) THEN
      planes.exited.successfully = planes.exited.successfully + 1 :GOTO
improper.place.continue
    END IF
  END IF
END IF

IF plane.data(15,k) = -5 OR plane.data(15,k) = -6 THEN
  IF cur.x.pos+cur.y.pos >= aw0422y1-20 AND cur.x.pos+cur.y.pos <= aw0422y1+20
THEN
    IF plane.data(12,k)=jetway.check.data(ABS(plane.data(15,k))) THEN
```

```
      planes.exited.successfully = planes.exited.successfully + 1 :GOTO
improper.place.continue
    END IF
  END IF
END IF

IF plane.data(15,k) = -7 OR plane.data(15,k) = -8 THEN
  IF cur.x.pos-cur.y.pos >= aw1331x1-20 AND cur.x.pos-cur.y.pos <= aw1331x1+20
THEN
    IF plane.data(12,k)=jetway.check.data(ABS(plane.data(15,k))) THEN
      planes.exited.successfully = planes.exited.successfully + 1 :GOTO
improper.place.continue
    END IF
  END IF
END IF

REM he was supposed to land, not exit
planes.exited.bad.hdg = planes.exited.bad.hdg + 1
special.message$ = "-- EXITTED AT IM PROP ER HEADING --"
GOTO improper.place.continue

improper.place.continue:
t$ = depart.msg$(INT(4*RND)+1)
tts$ = carrier.names$(plane.data(3,k))+STR$(plane.data(4,k))+" IS
GONE"+t$+special.message$
voice.type = k
GOSUB speakit
gone.exit:
GOSUB paint.it.white
FOR jj = 0 TO plane.params: plane.data(jj,k) = 0: NEXT   :'clear out info for
this plane

RETURN

REM  **********************************************
write.modes1:   'rewrite mode settings on dialog display

CALL MOVETO (263,10)
DrawText! "on  off"
IF params(1)=0 THEN bstate = 1 ELSE bstate = 2
BUTTON 32,bstate," ",(260,12)-(275,27),2
IF params(1)=0 THEN bstate=2 ELSE bstate = 1
BUTTON 33,bstate,"  DEMONSTRATION MODE",(285,12)-(508,27),2
IF params(2)=0 THEN bstate = 1 ELSE bstate = 2
BUTTON 34,bstate," ",(260,28)-(275,43),2
IF params(2)=0 THEN bstate=2 ELSE bstate = 1
BUTTON 35,bstate,"  AUTOMATIC TURN AT AIRWAYS",(285,28)-(508,43),2
IF params(3)=0 THEN bstate = 1 ELSE bstate = 2
BUTTON 36,bstate," ",(260,44)-(275,59),2
IF params(3)=0 THEN bstate=2 ELSE bstate = 1
BUTTON 37,bstate,"  COMPUTER FAILURE MODE",(285,44)-(508,59),2
IF params(4)=0 THEN bstate = 1 ELSE bstate = 2
BUTTON 38,bstate," ",(260,60)-(275,75),2
IF params(4)=0 THEN bstate=2 ELSE bstate = 1
BUTTON 39,bstate,"  VOICE",(285,60)-(508,75),2
```

```
IF params(14)=0 THEN bstate = 1 ELSE bstate = 2
BUTTON 40,bstate," ",(260,76)-(275,91),2
IF params(14)=0 THEN bstate=2 ELSE bstate = 1
BUTTON 41,bstate,"  CONFLICT ADVISORY",(285,76)-(437,91),2
CALL MOVETO (443,88): DrawText! "cycles"
x1=482: y1=79: x2=x1+22: y2=y1+10
EDIT FIELD 33,RIGHT$(STR$(adv.cycles),LEN(STR$(adv.cycles))-1),(x1,y1)-
(x2,y2),edit.fld.type


RETURN

REM  ***********************************************
write.modes2:   'rewrite level settings on dialog display

CALL MOVETO (260,108)
DrawText! "SKILL LEVEL"
bstate1=1: bstate2=1: bstate3=1: bstate4=1: bstate5=1
IF params(5)=1 THEN bstate1=2
IF params(5)=2 THEN bstate2=2
IF params(5)=3 THEN bstate3=2
IF params(5)=4 THEN bstate4=2
IF params(5)=5 THEN bstate5=2
BUTTON 42,bstate1,"1",(335,97)-(365,110),3
BUTTON 43,bstate2,"2",(370,97)-(400,110),3
BUTTON 44,bstate3,"3",(405,97)-(435,110),3
BUTTON 45,bstate4,"4",(440,97)-(470,110),3
BUTTON 46,bstate5,"5",(475,97)-(505,110),3
CALL MOVETO (332,120)
DrawText! "hard                  easy"

CALL MOVETO (260,143)
DrawText! "VOICE SPEED"
bstate1=1: bstate2=1: bstate3=1: bstate4=1: bstate5=1
IF params(8)=1 THEN bstate1=2
IF params(8)=2 THEN bstate2=2
IF params(8)=3 THEN bstate3=2
IF params(8)=4 THEN bstate4=2
IF params(8)=5 THEN bstate5=2
BUTTON 47,bstate1,"1",(335,132)-(365,145),3
BUTTON 48,bstate2,"2",(370,132)-(400,145),3
BUTTON 49,bstate3,"3",(405,132)-(435,145),3
BUTTON 50,bstate4,"4",(440,132)-(470,145),3
BUTTON 51,bstate5,"5",(475,132)-(505,145),3
CALL MOVETO (332,155)
DrawText! "slow                  fast"

CALL MOVETO (260,178)
DrawText! "VOICE LEVEL"
bstate1=1: bstate2=1: bstate3=1
IF params(7)=1 THEN bstate1=2
IF params(7)=2 THEN bstate2=2
IF params(7)=3 THEN bstate3=2
BUTTON 52,bstate1,"low",(335,167)-(385,182),3
BUTTON 53,bstate2,"med",(390,167)-(440,182),3
BUTTON 54,bstate3,"high",(445,167)-(495,182),3

CALL MOVETO (260,203)
DrawText! "POSITION"
bstate1=1: bstate2=1: bstate3=1
IF params(6)=1 THEN bstate1=2
IF params(6)=2 THEN bstate2=2
IF params(6)=3 THEN bstate3=2
BUTTON 55,bstate1,"mix",(314,192)-(355,207),3
BUTTON 56,bstate2,"ARTCC",(361,192)-(421,207),3
BUTTON 57,bstate3,"approach",(424,192)-(510,207),3

RETURN

REM  ***********************************************
write.modes3:   'rewrite count settings on dialog display

CALL MOVETO (260,233)
DrawText! "Planes to start with"
EDIT FIELD 30,RIGHT$(STR$(params(9)),LEN(STR$(params(9)))-1),(390,223)-
(416,233),edit.fld.type
CALL MOVETO (421,233)
DrawText! " or"
IF params(10) = 0 THEN bstate = 1 ELSE bstate = 2
BUTTON 58,bstate," ",(440,223)-(470,235),2
CALL MOVETO (464,233)
DrawText! "random"

CALL MOVETO (260,258)
DrawText! "Planes in your shift"
EDIT FIELD 31,RIGHT$(STR$(params(11)),LEN(STR$(params(11)))-1),(390,248)-
(416,258),edit.fld.type
CALL MOVETO (421,258)
DrawText! " or"
IF params(12) = 0 THEN bstate = 1 ELSE bstate = 2
BUTTON 59,bstate," ",(440,248)-(470,260),2
CALL MOVETO (464,258)
DrawText! "random"

CALL MOVETO (260,283)
DrawText! "Seconds between update cycles"
EDIT FIELD 32,RIGHT$(STR$(params(13)),LEN(STR$(params(13)))-1),(444,273)-
(470,283),edit.fld.type
EDIT FIELD 34,"Set any options and click the CONTINUE button to return to the
simulation.",(260,292)-(500,314),edit.fld.type

RETURN
```