1/23

UR1985_33

1985/33. DORIS - A drill log record information system

R.G. Richardson

Abstract

The program suite described is used to provide an index to the Department of Mines collection of geological drill logs. Programs are included for adding entries to the data-base and searching the data-base. The data are entered in a simple fill-in-the-spaces manner.

USING THE PROGRAMS

The programs are run on the Geological Survey Perkin-Elmer mini-computer, and it is assumed that the user is familiar with the standard data editing facilities.

Data-base maintenance

Data are put into a standard format (Appendix 1) before entry. Input is commenced by typing LOGADD and continues until 0 is typed in response to the reference number prompt. The drill type numbers are entered as either one or two digits on the same line.

At the end of input, a proof-sheet is output and the data is saved in file LOGTEMP.TMP. Any corrections should be made to this file using the standard editing facilities and a new proof sheet then printed and checked.

The new data in LOGTEMP.TMP must be added to the end of the existing data-base by typing LOGMERGE. To ensure that the file remains in reference number order, it should be sorted periodically using the LOGSORT command. As sorting is a slow process it should not be carried out more often than necessary.

Corrections to the main data-base would normally be made by deleting entries using the system editor in hexadecimal mode and then putting the corrected entries in using LOGADD.

Data-base searching

The search phase is entered by typing LOGSRCH. An entry will be retrieved only if one value in each search option specified is found.

The program requests the following data:

(i) Locality - one or more localities on a single line, each locality terminated by /. If a search by locality is not required leave the line blank.

e.g. FINGAL/AVOCA/SMITHTON/

(ii) Quadrangle - one or more quadrangle numbers in the format used for location.

(iii) Map sheet - one or more 1:50 000 map sheet numbers in the format used for location.

(iv) Search by area - leave blank for no search by area.
- Y for finding entries inside a specified area. Entries on the boundary of the area will not be found. The program prompts for the number of vertices. If 2 is entered a simple rectangle parallel to the grid may be entered by specifying the co-ordinates of any two diagonally opposite corners. Otherwise the program prompts for the vertex co-ordinates. Co-ordinates should be in the same form as used on the input sheets.

(v) Purpose - one or more purpose numbers in the same format used for location.

(vi) Core held - enter Y if only holes for which core is held are to be retrieved.
Leave blank otherwise.

(vii) E.L. or M.L. - one or more E.L. or M.L. numbers on a single line, each terminated by ;. Leave blank if a search by E.L. or M.L. is not required.

e.g. 5/63;123P/M;16/83;

(viii) Organisation - details as for location.

(ix) Drilled by - details as for purpose.

(x) Range of commencement dates - enter the earliest and latest possible commencement dates at the prompt in the form used on the input sheet. Leave blank if this option is not required.

```
e.g. 12/1876
     3/1878
```

(xi) Drill type - one or more drill type numbers on a single line, each group terminated by /. Leave blank if not required.

```
e.g. 0/2/ diamond or rotary
     02/4/ diamond and rotary or auger.
```

(xii) Geophysical logging - enter Y if only holes that have been geophysically logged are to be retrieved. Leave blank otherwise.

THE PROGRAMS

LOGADD (Appendix 2)

This program accepts data from the keyboard and copies it to a file in a format suitable for editing. Prompts are used to guide the user. The file created (LOGTEMP.TMP) is then edited using the standard edit facilities.

LOGMERGE (Appendix 3)

The data from LOGTEMP.TMP are converted to the format of the data-base. The data-base is copied to a temporary work file and the new data are added to the end of this. The combined file is then copied back to the original data-base file.

LOGSORT (Appendix 4)

The data are sorted into ascending reference number order and written to a temporary file in the new order. The temporary file is then copied back to the original data-base file.

LOGSRCH (Appendix 5)

The data-base is searched for the occurrence of specified strings and a point-in-polygon algorithm is used to locate data from within a specified area.

[19 June 1985]

APPENDIX 1

BOREHOLE LOG DATA SHEET

REFERENCE NO: _ _ _ _ _ (5)

BOREHOLE NO: _ _ _ _ _ _ _ _ _ _ _ _ (12)

LOCALITY: _ (20)

QUADRANGLE: _ _ (2)

1:50 000 MAP SHEET _ _ _ _ _ (5)

AMG MAP REFERENCE: _ _ _ _ 5 _ _ _ _ (9)

PURPOSE:

0 ☐ Engineering geology	1 ☐ Metallic minerals	2 ☐ Non-metallic minerals
3 ☐ Fuels	4 ☐ Stratigraphic	(Tick one)

PLAN NO: _ _ _ _ _ _ (6)

REPORT: _ _ _ _ _ _ _ _ _ _ _ _ (12)

LOG LOCATION: _ _ _ (3)

CORE HELD: Y/N (1)

E.L. OR M.L. etc. _ _ _ _ _ _ _ (7)

ORGANISATION: _ (20)

DRILLED BY:

0 ☐ Department of Mines	1 ☐ Hydro-electric Commission
2 ☐ Commonwealth	3 ☐ Private contractor (Tick one)

DATE COMMENCED: _ _ / _ _ _ _ (7)

DEPTH: _ _ _ _ (4)

DRILL TYPE :

0 ☐ Diamond	1 ☐ Cable Tool	2 ☐ Rotary
3 ☐ Percussion	4 ☐ Auger	(tick two maximum)

GEOPHYSICAL LOGGING: Y/N (1)

APPENDIX 2

Program LOGADD

```
*LOGADD.CSS
*FOR RUNNING LOGADD AND PRINTING A PROOF SHEET
$IFX LOGTEMP.TMP; $WRITE ** CHECK LAST LOGMERGE; $EXIT; $ENDC
PRE ETM; AL LOGTEMP.TMP,IN,116
* SET UP THE OUTPUT FILE
L LOGADD,3; AS 6,LOGTEMP.TMP; * LOAD PROG AND SET LU 6
ST
PRINT LOGTEMP.TMP
$WRITE  FILE IS LOGTEMP.TMP; ENA ETM; $EXIT
```

```
$TITL LOGADD.FTN - ADD LOGS TO THE LOG INDEX
C TAKES KEYBOARD INPUT AND PREPARES A PROOF SHEET
      CHARACTER*12 BHOLE,REPORT
      CHARACTER*20 LOCAL,ORGAN
      CHARACTER*5 SHEET
      CHARACTER*1 PURP,CORE,DRILR,LOGGED
      CHARACTER*6 PLANNO
      CHARACTER*3 LOGLOC
      CHARACTER*7 ELML,DATE
      CHARACTER*2 TYPE,IQUAD
      INTEGER*2 IREF,IDEEP
      INTEGER*4 MAPREF
C
C INPUT IS REFERENCE NO.,BOREHOLE NO.,LOCALITY,QUADRANGLE,MAP SHEET,
C             MAP REFERENCE,PURPOSE,PLAN NO.,REPORT,LOG LOCATION,
C             CORE HELD,EL OR ML,ORGANISATION,DRILLED BY,DATE COMMENCED,
C             DEPTH,DRILL TYPE,GEOPHYSICAL LOGGING
C
      OPEN (UNIT=5,FILE='CON:')
C OPEN THE CONSOLE FOR INPUT.  ASSUME THAT OUTPUT FILE 6 HAS BEEN SET
C UP ELSEWHERE
C
   20 WRITE(5,100)
  100 FORMAT(' REF NO (0 TO END)')
      READ(5,*) IREF
      IF (IREF .EQ. 0) GOTO 50
C END OF BATCH SO CLOSE OFF
      WRITE(5,101)
  101 FORMAT(' BOREHOLE NO')
      READ(5,201) BHOLE
  201 FORMAT(A12)
      WRITE(5,102)
  102 FORMAT(' LOCALITY')
      READ(5,202) LOCAL
  202 FORMAT(A20)
      WRITE(5,103)
  103 FORMAT(' QUADRANGLE')
      READ(5,208) IQUAD
      WRITE(5,104)
  104 FORMAT(' MAP SHEET')
      READ(5,203) SHEET
  203 FORMAT(A5)
      WRITE(5,105)
  105 FORMAT(' MAP REF')
      READ(5,*) MAPREF
      WRITE(5,106)
  106 FORMAT(' PURPOSE')
      READ(5,204) PURP
  204 FORMAT(A1)
      WRITE(5,107)
  107 FORMAT(' PLAN NO')
      READ(5,205) PLANNO
  205 FORMAT(A6)
      WRITE(5,108)
  108 FORMAT(' REPORT')
      READ(5,201) REPORT
      WRITE(5,109)
  109 FORMAT(' LOG LOCN')
      READ(5,206) LOGLOC
  206 FORMAT(A3)
      WRITE(5,110)
  110 FORMAT(' CORE HELD')
      READ(5,204) CORE
      WRITE(5,111)
  111 FORMAT(' EL OR ML')
      READ(5,207) ELML
  207 FORMAT(A7)
      WRITE(5,112)
  112 FORMAT(' ORGANISATION')
      READ(5,202) ORGAN
      WRITE(5,113)
  113 FORMAT(' DRILLED BY')
      READ(5,204) DRILR
      WRITE(5,114)
  114 FORMAT(' DATE COMMENCED')
      READ(5,207) DATE
      WRITE(5,115)
  115 FORMAT(' DEPTH')
      READ(5,*) IDEEP
      WRITE(5,116)
  116 FORMAT(' DRILL TYPE(S)')
      READ(5,208) TYPE
  208 FORMAT(A2)
      WRITE(5,117)
  117 FORMAT(' GEOPHYSICAL LOGGING')
      READ(5,204) LOGGED
C READ ONE DAT SHEET
C
C SO WRITE TO THE OUTPUT FILE
       WRITE(6,300) IREF,BHOLE,LOCAL,IQUAD,SHEET,MAPREF,PURP,PLANNO,
     .             REPORT,LOGLOC,CORE,ELML,ORGAN,DRILR,DATE,IDEEP,
     .             TYPE,LOGGED
  300 FORMAT(I6,'@',A12,'@',A20,'@',A2,'@',A5,'@',I9,'@',A1,'@',A6,'@',
     .       A12,'@',A3,'@',A1,'@',A7,'@',A20/1X,A1,'@',A7,'@',I4,'@',
     .       A2,'@',A1/1X)
C WRITE WITH DELIMITERS AFTER FIELD AND A BLANK AFTER EACH ENTRY
      GOTO 20
C BACK AROUND
C
C OHTERWISE STOP
   50 CLOSE(UNIT=6,STATUS='KEEP')
      STOP
      END
```

APPENDIX 3

Program LOGMERGE

```
*LOGMERGE.CSS
* FOR MERGING CORRECTED FILE WITH MAIN FILE
PRE ETM; XDE MRGTMP.TMP; *DELETE SCRATCH FILE
AL MRGTMP.TMP,IN,108; L LOGMERGE,10; * AL SCRATCH FILE AND LOAD PROG
REP LOGTEMP.TMP,FF00; AS 6,LOGTEMP.TMP,ERO; AS 4,MRGTMP.TMP; ST
$IFNE 0; $WRITE TRANSLATE ERROR; ENA ETM; $EXIT; $ENDC
REP MRGTMP.TMP,FF00
$BUILD COPY.CMD
IN LOGIND.DAT
AL SYSF:TEMP.DAT,IN,108/6/5
OUT SYSF:TEMP.DAT
COPY *,*
IN MRGTMP.TMP
COPY *,*
END
$ENDB
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE COPY-MERGE ERROR; ENA ETM; $EXIT; $ENDC
REP LOGIND.DAT,0; REP SYSF:TEMP.DAT,FF00
DE COPY.CMD,LOGIND.DAT
$BUILD COPY.CMD
IN SYSF:TEMP.DAT
AL LOGIND.DAT,IN,108/10/3
OUT LOGIND.DAT
COPY *,*
REW I
REW O
VERIFY *,*
END
$ENDB
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE COPY BACK FAILED; ENA ETM; $EXIT; $ENDC
REP LOGIND.DAT,FF00; REP SYSF:TEMP.DAT,0; REP LOGTEMP.TMP,0
REP MRGTMP.TMP,0
DE COPY.CMD,MRGTMP.TMP,LOGTEMP.TMP,SYSF:TEMP.DAT
ENA ETM
$EXIT
```

```
$TITL LOGMERGE.FTN - CONVERT PROOF FILE TO MASTER FILE FORMAT
C
      INTEGER*4 NORTH
      INTEGER*2 EAST,IMNTH,IYR
C SPLIT MAPREF AND DATE INTO RELEVANT PARTS
      CHARACTER*12 BHOLE,REPORT
      CHARACTER*20 LOCAL,ORGAN
      CHARACTER*5 SHEET
      CHARACTER*1 PURP,CORE,DRILR,LOGGED
      CHARACTER*6 PLANNO
      CHARACTER*3 LOGLOC
      CHARACTER*7 ELML,DATE
      CHARACTER*2 TYPE,IQUAD
      INTEGER*2 IREF,IDEEP
      INTEGER*4 MAPREF
C
      CHARACTER*1 BUFF(108)
C BUFFER ARRAYTO ALLOW BINARY WRITE
      EQUIVALENCE (BUFF(1),LOCAL),(BUFF(21),ORGAN),(BUFF(41),BHOLE),
     .     (BUFF(53),REPORT),(BUFF(65),ELML),
     .     (BUFF(72),PLANNO),(BUFF(78),SHEET),(BUFF(83),NORTH),
     .     (BUFF(87),LOGLOC),(BUFF(90),TYPE),(BUFF(93),IREF),
     .     (BUFF(95),IMNTH),(BUFF(97),IYR),
     .     (BUFF(99),IQUAD),(BUFF(101),IDEEP),(BUFF(103),EAST),
     .     (BUFF(105),PURP),(BUFF(106),CORE),(BUFF(107),DRILR),
     .     (BUFF(108),LOGGED)
C USED FOR BINARY OUTPUT FOR FASTER ACCESS
C
      OPEN(UNIT=4,FORM='BINARY',RECL=108)
C OPEN THE OUTPUT FILE FOR BINARY OPERATIONS
   10  READ(6,300,END=20) IREF,BHOLE,LOCAL,IQUAD,SHEET,EAST,NORTH,PURP,
     .      PLANNO,REPORT,LOGLOC,CORE,ELML,ORGAN,DRILR,IMNTH,IYR,IDEEP,
     .      TYPE,LOGGED
  300 FORMAT(I6,1X,A12,1X,A20,1X,A2,1X,A5,1X,I4,I5,1X,A1,1X,A6,1X,
     .       A12,1X,A3,1X,A1,1X,A7,1X,A20/1X,A1,1X,I2,1X,I4,1X,I4,1X,
     .       A2,1X,A1/1X)
      WRITE(4) BUFF
C BINARY WRITE TO OUTPUT FILE
      GOTO 10
C BACK ROUND
C
C AT END
   20 CLOSE(UNIT=4,STATUS='KEEP')
      CLOSE(UNIT=6,STATUS='KEEP')
      END
```

APPENDIX 4

Program LOGSORT

```
*LOGSORT.CSS
* FOR SORTING DRILL LOG FILE INTO REFERENCE NO. ORDER
PRE ETM
L LOGSORT,10; AS 4,LOGIND.DAT,ERO; AL SYSF:TEMP.DAT,IN,108/5/2
AS 6,SYSF:TEMP.DAT; ST
$IFNE 0; $WRITE SORT ERROR; ENA ETM; $EXIT; $ENDC
$BUILD COPY.CMD
IN SYSF:TEMP.DAT
AL LOGIND.DAT,IN,108/10/5
OUT LOGIND.DAT
COPY *,*
REW I
REW O
VERIFY *,*
END
$ENDB
REP SYSF:TEMP.DAT,FF00; REP LOGIND.DAT,0; DE LOGIND.DAT
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE SORT-COPY ERROR; ENA ETM; $EXIT; $ENDC
REP LOGIND.DAT,FF00; REP SYSF:TEMP.DAT,0; DE SYSF:TEMP.DAT,COPY.CMD
ENA ETM; $EXIT
```

```
$TITL LOGSORT.FTN - SORT THE DRILL LOG FILE INTO ASCENDING REF NO ORDER
C
      INTEGER*4 NORTH
      INTEGER*2 EAST,IMNTH,IYR
C SPLIT MAPREF AND DATE INTO RELEVANT PARTS
      CHARACTER*12 BHOLE,REPORT
      CHARACTER*20 LOCAL,ORGAN
      CHARACTER*5 SHEET
      CHARACTER*1 PURP,CORE,DRILR,LOGGED
      CHARACTER*6 PLANNO
      CHARACTER*3 LOGLOC
      CHARACTER*7 ELML,DATE
      CHARACTER*2 TYPE,IQUAD
      INTEGER*2 IREF,IDEEP
C THE FOLLOWING ARE FOR THE SORT
      INTEGER*2 INDEX(8000),IA(8000)
C ALLOW FOR UP TO 8000 ENTRIES
C
      CHARACTER*1 BUFF(108)
C BUFFER ARRAYTO ALLOW BINARY WRITE
      EQUIVALENCE (BUFF(1),LOCAL),(BUFF(21),ORGAN),(BUFF(41),BHOLE),
     .     (BUFF(53),REPORT),(BUFF(65),ELML),
     .     (BUFF(72),PLANNO),(BUFF(78),SHEET),(BUFF(83),NORTH),
     .     (BUFF(87),LOGLOC),(BUFF(90),TYPE),(BUFF(93),IREF),
     .     (BUFF(95),IMNTH),(BUFF(97),IYR),
     .     (BUFF(99),IQUAD),(BUFF(101),IDEEP),(BUFF(103),EAST),
     .     (BUFF(105),PURP),(BUFF(106),CORE),(BUFF(107),DRILR),
     .     (BUFF(108),LOGGED)
C USED FOR BINARY OUTPUT FOR FASTER ACCESS
C
      OPEN(UNIT=4,FORM='BINARY',ACCESS='DIRECT',RECL=108)
C OPEN THE INPUT FILE FOR RANDOM AND SEQUENTIAL ACCESS
      REWIND 4
      IREC=0
C COUNTER FOR NUMBER OF RECORDS
   10 READ(4,END=20) BUFF
      IREC=IREC+1
      IF (IREC .GT. 8000) STOP 'MORE THAN 8000 ENTRIES'
      INDEX(IREC)=IREC
C FILL SORT INDEX
      IA(IREC)=IREF
      GOTO 10
C BACK AROUND
C
C NOW FOR THE WORK
   20 CONTINUE
      IF (IREC .EQ. 0) STOP 'NO RECORDS'
      CALL BUBSTI(INDEX,IA,1,IREC)
C SORT INTO ASCENDING NUMBER - I.E. CHRON ORDER
      OPEN(UNIT=6,FORM='BINARY',RECL=108)
C OPEN OUTPUT FILE
      DO 30 I=1,IREC
      READ(4,REC=INDEX(I)) BUFF
      WRITE(6) BUFF
C COPY TO NEW ORDER
   30 CONTINUE
      CLOSE(UNIT=6,STATUS='KEEP')
      CLOSE(UNIT=4,STATUS='KEEP')
      END
      SUBROUTINE BUBSTI(IR,IA,IBASE,N)
      INTEGER*2 IA(N)
      INTEGER*2 IR(N)
      LOGICAL NSWAP
      IF (N .LE. 1) RETURN
C NOTHING TO SORT
      NM1=N-1
      DO 30 J=IBASE,NM1
      NSWAP=.TRUE.
      IRI=IR(1)
      DO 40 I=IBASE,NM1
      IP1=I+1
      IRIP1=IR(IP1)
      IF (IA(IRI) .LE. IA(IRIP1)) GOTO 40
      NSWAP=.FALSE.
      IR(I)=IRIP1
      IR(IP1)=IRI
      IRIP1=IRI
   40 IRI=IRIP1
      IF (NSWAP) RETURN
   30 CONTINUE
      RETURN
      END
```

APPENDIX 5

Program LOGSRCH

```
*LOGSRCH.CSS - SEARCH THE DRILL LOG FILE
L LOGSRCH,10; AS 4,LOGIND.DAT,ERO; XAL SYSF:LOGIND.TMP,IN,132/3/2
AS 6,SYSF:LOGIND.TMP; REW 6; AS 5,CON:; TEMPFILE 3,IN,80; ST
$IFX SYSF:LOGIND.TMP; PRI SYSF:LOGIND.TMP,DEL; $ENDC; $EXIT
```

```
$TITL  LOGSRCH.FTN - SEARCH DRILL LOG FILE
C
      INTEGER*4 NORTH
      INTEGER*2 EAST,IMNTH,IYR
C SPLIT MAPREF AND DATE INTO RELEVANT PARTS
      CHARACTER*12 BHOLE,REPORT
      CHARACTER*20 LOCAL,ORGAN
      CHARACTER*5 SHEET
      CHARACTER*1 PURP,CORE,DRILR,LOGGED
      CHARACTER*6 PLANNO
      CHARACTER*3 LOGLOC
      CHARACTER*7 ELML,DATE,SDATE,EDATE
      CHARACTER*2 TYPE,IQUAD
      INTEGER*2 IREF,IDEEP,SMNTH,SYR,EMNTH,EYR
C
      CHARACTER*1 BUFF(108)
C BUFFER ARRAYTO ALLOW BINARY WRITE
      EQUIVALENCE (BUFF(1),LOCAL),(BUFF(21),ORGAN),(BUFF(41),BHOLE),
     .     (BUFF(53),REPORT),(BUFF(65),ELML),
     .     (BUFF(72),PLANNO),(BUFF(78),SHEET),(BUFF(83),NORTH),
     .     (BUFF(87),LOGLOC),(BUFF(90),TYPE),(BUFF(93),IREF),
     .     (BUFF(95),IMNTH),(BUFF(97),IYR),
     .     (BUFF(99),IQUAD),(BUFF(101),IDEEP),(BUFF(103),EAST),
     .     (BUFF(105),PURP),(BUFF(106),CORE),(BUFF(107),DRILR),
     .     (BUFF(108),LOGGED)
C USED FOR BINARY OUTPUT FOR FASTER ACCESS
C
      CHARACTER*80 SQUAD,SLOCN,SMAP,SPURP,SELML,SORGAN,SDRILL,STYPE
      CHARACTER*1 SCORE,SLOGD
      INTEGER*2 FFEED,ITYPE
C USED TO PUT FORM FEEDS INTO OUTPUT FILE
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100)
      INTEGER*2 SWATH(100,25)
C ABOVE 2 LINES USED FOR FSTJDN POINT IN POLYGON SEARCH
      LOGICAL NOTIN,INOUT,SEMIN
      INTEGER*4 FCOUNT
      DATA FFEED/3072/
C THE FORM FEED CHARACTER IN A1
      OPEN(UNIT=4,FORM='BINARY',RECL=108)
C OPEN THE INPUT FILE
      WRITE(5,100)
  100 FORMAT(' SEARCHING DRILL LOG INDEX')
C NOW FOR A FULL SEARCH
      WRITE(6,400)
  400 FORMAT(' SEARCH OPTIONS ARE:')
      WRITE(5,101)
  101 FORMAT(' FOR EACH OPTION ENTER EITHER A BLANK LINE OR'/
     . ' THE DESIRED VALUES FOR THE OPTION, EACH TERMINATED BY'
     . ,' A /''/' E.G.  FRED/BILL/TOM/SMITH/')
      WRITE(5,102)
  102 FORMAT(' LOCALITY?')
      READ(5,201) SLOCN
      IF (SLOCN(1:1) .NE. ' ') WRITE (6,402) SLOCN
  402 FORMAT(' LOCALITY',10X,A80)
      WRITE(5,103)
  103 FORMAT(' QUADRANGLE?')
      READ(5,201) SQUAD
  201 FORMAT(A80)
      IF (SQUAD(1:1) .NE. ' ') WRITE(6,401) SQUAD
  401 FORMAT(' QUADRANGLE',10X,A80)
      WRITE(5,106)
  106 FORMAT(' MAP SHEET?')
      READ(5,201) SMAP
      IF (SMAP(1:1) .NE. ' ') WRITE(6,405) SMAP
  405 FORMAT(' MAP SHEET',10X,A80)
      WRITE(5,108)
  108 FORMAT(' SEARCH FOR SITES WITHIN A SPECIFIED AREA (Y)?')
      READ(5,200) ITYPE
  200 FORMAT(A1)
      IF (ITYPE .EQ. 'Y') GOTO 23
C WANT TO READ COORDS
      NCOORD=0
      GOTO 26
C DONT WANT TO SEARCH FOR SITES IN AN AREA
C
   23 CONTINUE
C READ VERTICES
      WRITE(5,109)
  109 FORMAT(' NUMBER OF VERTICES OF AREA?'/
     .   ' ENTER 2 TO SPECIFY A RECTANGLE')
      READ(5,*) NCOORD
      IF (NCOORD .LT. 2) GOTO 23
C NEED AT LEAST 3 VERTICES TO DEFINE AN AREA
      IF (NCOORD .EQ. 2) THEN
      WRITE(5,515)
  515 FORMAT(' ENTER COORDS OF TWO DIAGONALLY OPPOSITE CORNERS OF ',
     .   'THE RECTANGLE'/' ONE PER LINE IN THE FORM EASTNORTH')
      ELSE
      WRITE(5,110)
  110 FORMAT(' ENTER VERTEX COORDS IN CLOCKWISE ORDER'/
     .' ONE PER LINE IN FORM EASTNORTH')
      ENDIF
      IXMAX=-999999999
      IXMIN=-IXMAX
      IYMAX=IXMAX
      IYMIN=IXMIN
C USE THESE TO STORE THE RANGE OF THE AREA
      DO 24 I=1,NCOORD
      READ(5,303) IX,IY
  303 FORMAT(I4,I5)
C GET INTO 100'S OF METRES EAST AND NORTH
      IXMAX=MAX0(IX,IXMAX)
      IXMIN=MIN0(IX,IXMIN)
      IYMAX=MAX0(IY,IYMAX)
      IYMIN=MIN0(IY,IYMIN)
C FIND THE RANGE
      X(I)=IX*0.1
      Y(I)=IY*0.1
C CONVERT TO KM BEFORE STORING
   24 CONTINUE
      IF (NCOORD .GT. 2) THEN
      IF (X(1) .NE. X(NCOORD) .OR. Y(1) .NE. Y(NCOORD)) THEN
       NCOORD=NCOORD+1
       X(NCOORD)=X(1)
       Y(NCOORD)=Y(1)
C CLOSE AREA
      ENDIF
      WRITE(6,407)
  407 FORMAT(' AREA COORDINATES (KM)')
```

```
      WRITE(6,408) (X(I),Y(I),I=1,NCOORD)
  408 FORMAT(20X,4(F14.2,F9.2))
      NCOORD=NCOORD-1
C ROUTINE FOR LOCATING POINTS DOESNT REQUIRE CLOSURE
      CALL PREPLY(X,Y,NCOORD,YINTVL,INTVLS,SWATH,RSLOPE)
C CUT THE SEARCH AREA UP INTO SWATHS
      ELSE
      WRITE(6,516) IXMIN,IXMAX,IYMIN,IYMAX
  516 FORMAT(' SEARCH RECTANGLE FROM',I5,'00ME TO',I5,'00ME AND FROM',
     .       I6,'00MN TO',I6,'00MN')
      ENDIF
C
   26 CONTINUE
      WRITE(5,111)
  111 FORMAT(' PURPOSE (0-4)?')
      READ(5,201) SPURP
      IF(SPURP(1:1) .NE. ' ') WRITE(6,409) SPURP
  409 FORMAT(' PURPOSE',10X,A80)
      WRITE(5,112)
  112 FORMAT(' MUST CORE BE HELD?')
      READ(5,202) SCORE
  202 FORMAT(A1)
      IF (SCORE(1:1) .EQ. 'Y') WRITE(6,410)
  410 FORMAT(' CORE MUST BE HELD')
      WRITE(5,114)
  114 FORMAT(' E.L. OR M.L. (USE ; TO SEPARATE OPTIONS HERE)?')
      READ(5,201) SELML
      IF (SELML(1:1) .NE. ' ') WRITE(6,411) SELML
  411 FORMAT(' E.L. OR M.L.',10X,A80)
      WRITE(5,115)
  115 FORMAT(' ORGANISATION?')
      READ(5,201) SORGAN
      IF (SORGAN(1:1) .NE. ' ') WRITE(6,412) SORGAN
  412 FORMAT(' ORGANISATION',10X,A80)
      WRITE(5,116)
  116 FORMAT(' DRILLED BY (0-3)?')
      READ(5,201) SDRILL
      IF (SDRILL(1:1) .NE. ' ') WRITE(6,413) SDRILL
  413 FORMAT(' DRILLED BY',10X,A80)
      WRITE(5,117)
  117 FORMAT(' RANGE OF COMMENCEMENT DATES'/
     .    ' ENTER EARLIEST DATE IN FORM MM/YYYY')
      READ(5,203) SDATE
  203 FORMAT(A7)
      IF (SDATE(2:2) .NE. ' ') THEN
      WRITE(5,118)
  118 FORMAT(' ENTER LATEST DATE IN FORM MM/YYYY')
      READ(5,203) EDATE
      WRITE(6,414) SDATE,EDATE
  414 FORMAT(' LOOKING FOR DRILL HOLES COMMENCING BETWEEN ',A7,' AND ',
     . A7)
      CALL DATIN(SMNTH,SYR,SDATE)
      CALL DATIN(EMNTH,EYR,EDATE)
      ENDIF
      WRITE(5,119)
  119 FORMAT(' DRILL TYPE (0-4)?')
      READ(5,201) STYPE
      IF (STYPE(1:1) .NE. ' ') WRITE(6,415) STYPE
  415 FORMAT(' DRILL TYPE',10X,A80)
      WRITE(5,120)
  120 FORMAT(' MUST HOLE BE GEOPHYSICALLY LOGGED?')
      READ(5,202) SLOGD
      IF (SLOGD(1:1) .NE. ' ') WRITE(6,416)
  416 FORMAT(' HOLE MUST BE GEOPHYSICALLY LOGGED')
C END OF OPTION INPUT
C
C*****************************************************************
C NOW FOR THE SEARCH
      FCOUNT=0
C COUNTER FOR NUMBER OF FOUND ENTRIES
   27 READ(4,END=40) BUFF
C READ ONE ENTRY
      IF (NOTIN(SMAP,SHEET)) GOTO 27
C NOT THE RIGHT SHEET
      IF(NOTIN(SLOCN,LOCAL)) GOTO 27
C NOT THE RIGHT LOCATION
      IF(NOTIN(SORGAN,ORGAN)) GOTO 27
C NOT THE RIGHT ORGANISATION
      IF (SCORE .EQ. 'Y' .AND. CORE .EQ. 'N') GOTO 27
C CORE NOT HELD
      IF (SLOGD .EQ. 'Y' .AND. LOGGED .EQ. 'N') GOTO 27
C NOT GEOPHYSICALLY LOGGED
      IF (NCOORD .NE. 0) THEN
       IF (EAST .LT. IXMIN .OR. EAST .GT. IXMAX) GOTO 27
       IF (NORTH .LT. IYMIN .OR. NORTH .GT. IYMAX) GOTO 27
C CANT POSSIBLY BE IN AREA
      IF (NCOORD .GT. 2) THEN
       XTMP=EAST*0.1
       YTMP=NORTH*0.1
C CONVERT TO KM
       IF (.NOT. INOUT(XTMP,YTMP,X,Y,YINTVL,INTVLS,SWATH,RSLOPE))
     . GOTO 27
C NOT IN THE SPECIFIED AREA
      ENDIF
C FOR THE SIMPLE RECTANGLE DONT NEED THE POLYGON SEARCH
      ENDIF
      IF (SDRILL(1:1) .NE. ' ' .AND. INDEX(SDRILL,DRILR) .EQ. 0) GOTO 27
C NOT THE DESIRED DRILLER
      IF (SPURP(1:1) .NE. ' ' .AND. INDEX(SPURP,PURP) .EQ. 0) GOTO 27
C NOT THE RIGHT PURPOSE
      IF (SQUAD(1:1) .NE. ' ' .AND. INDEX (SQUAD,IQUAD) .EQ. 0) GOTO 27
C NOT RIGHT QUADRANGLE
      IF (SDATE(2:2) .NE. ' ') THEN
      IF (IYR .LT. SYR .OR. IYR .EQ. SYR .AND. IMNTH .LT. SMNTH) GOTO 27
C EARLIER THAN THE FIRST DATE
      IF (IYR .GT. EYR .OR. IYR .EQ. EYR .AND. IMNTH .GT. EMNTH) GOTO 27
C LATER THAN THE LAST DATE
      ENDIF
      IF (NOTIN(STYPE,TYPE)) GOTO 27
C NOT RIGHT TYPE OF DRILL
      IF (SEMIN(SELML,ELML)) GOTO 27
C NOT RIGHT E.L. OR M.L.
C IF FINALLY GET TO HERE HAVE FOUND AN ENTRY THAT MATCHES THE
C SEARCH OPTIONS
C
      IF (MOD(FCOUNT,28) .EQ. 0) CALL HEAD
C HEAD PAGE
      FCOUNT=FCOUNT+1
C INCREMENT FOUND ENTRY COUNTER
C
```

```
      WRITE(6,417) IREF,BHOLE,LOCAL,IQUAD,SHEET,EAST,NORTH,PURP,
     .     PLANNO,REPORT,LOGLOC,CORE,ELML,ORGAN,DRILR,IMNTH,IYR,
     .     IDEEP,TYPE(1:1),TYPE(2:2),LOGGED
  417 FORMAT(2X,I5,'|',A12,'|',A20,'| ',A2,' |',A5,'|',I4,I5,'|',
     .        3X,A1,3X,'|',A6,'|',A12,'|',3X,A3,'   |   ',A1,' |',A7,' |'
     .        12X,'|',A20,'|',3X,A1,3X,'|',I2,'/',I4,'|',I5,'| ',
     .        2(A1,1X),'|',3X,A1,3X,'|')
      GOTO 27
C BACK AROUND
C
C NOW AT THE END OF THE RUN
   40 CONTINUE
      WRITE(5,121) FCOUNT
  121 FORMAT(I7,' RECORDS WERE FOUND'/' DO YOU WANT A',
     .        ' PRINTOUT (Y OR N)?')
      READ(5,200) ITYPE
      IF (ITYPE .NE. 'Y' .OR. FCOUNT .EQ. 0) THEN
       CLOSE(UNIT=6,STATUS='DELETE')
       STOP
C CLOSE FILE AND DELETE IT THEN STOP
      ENDIF
      CLOSE(UNIT=4,STATUS='KEEP')
C CLOSE THE INPUT FILE TO MAKE SPACE FOR THE NEXT FILE
      OPEN(UNIT=7,FILE='LOGNMON.TXT',SHARE='ERO')
C OPEN THE FILE OF EXPANSIONS OF ABBREVIATIONS
      WRITE(6,418) FFEED
  418 FORMAT(A1)
   45 READ(7,419,END=46) SQUAD
  419 FORMAT(A80)
      WRITE(6,419) SQUAD
      GOTO 45
C COPY THE ABBREV FILE
   46 CONTINUE
      CLOSE(UNIT=6,STATUS='KEEP')
      STOP
      END
      LOGICAL FUNCTION NOTIN(SSTRNG,RSTRNG)
C SSTRNG CONTAINS A SERIES OF STRINGS TERMINATED BY /
C E.G.   FRED/BILL/TOM/
C AND RSTRNG IS SEARCHED FOR THE OCCURENCE OF ONE OF THESE STRINGS
C IF ONE IS FOUND TO MATCH NOTIN IS .FALSE.
C IF STRNG IS EMPTY NOTIN IS .FALSE.
      CHARACTER*80 SSTRNG
      CHARACTER* (*) RSTRNG
      NOTIN=.FALSE.
      IF (SSTRNG(1:2) .EQ. '  ') RETURN
C NO NEED TO LOOK
      ILAST=1
C COUNTER FOR POSITION IN SSTRNG
   20 ICHAR=INDEX(SSTRNG(ILAST:80),'/')
C LOOK FOR / AS TERMINATOR OF STRING
      IF (ICHAR .EQ. 0) GOTO 25
C HAVENT FOUND A MATCH BY THE END OF THE LINE
      IF (INDEX(RSTRNG,SSTRNG(ILAST:ILAST+ICHAR-2)) .NE. 0) RETURN
C LOOK BETWEEN /S IN SSTRNG AND IF FIND A MATCH RETURN
      ILAST=ILAST+ICHAR
C POSITION AFTER THE /
      IF (ILAST .LE. 79) GOTO 20
C BACK ROUND IF NOT AT END OF LINE
C
```

```
C ELSE END
   25 CONTINUE
      NOTIN=.TRUE.
C NO MATCH
      RETURN
      END
      SUBROUTINE HEAD
C HEAD UP PAGES FROM SEARCH
      INTEGER*2 FFEED
      DATA FFEED/3072/
C USED TO PUT FORM FEEDS IN OUTPUT FILE
      WRITE(6,100) FFEED
  100 FORMAT(1A1/' REF NO|BOREHOLE NO |',6X,'LOCALITY',6X,'|QUAD',
     . '|SHEET| AMG REF |PURPOSE| PLAN |   REPORT   |LOG LOCN',
     . '|CORE|EL OR ML|'/12X,'|    ORGANISATION    |DRILLER|  DATE |',
     . 'DEPTH|DRILL|GEOPHYS|'/114('-'))
      RETURN
      END
      SUBROUTINE DATIN(IMN,IYR,DATE)
C USE TO READ MONTH AND YEAR FROM MM/YYYY FORMAT IN DATE
C REPLACE / WITH , IN 12/1976
      INTEGER*2 IMN,IYR
      CHARACTER*7 DATE
      DO 10 I=1,9
      IF (DATE(I:I) .EQ. '/') DATE(I:I)=','
   10 CONTINUE
      REWIND 3
      WRITE(3,115) DATE
  115 FORMAT(A7)
      REWIND 3
      READ(3,*) IMN,IYR
C PUT OUT TO TEMPORARY FILES AND READ BACK IN IN FREE FORMAT
      RETURN
      END
      LOGICAL FUNCTION SEMIN(SSTRNG,RSTRNG)
C SSTRNG CONTAINS A SERIES OF STRINGS TERMINATED BY ;
C E.G.    FRED;BILL;TOM;
C AND RSTRNG IS SEARCHED FOR THE OCCURENCE OF ONE OF THESE STRINGS
C IF ONE IS FOUND TO MATCH SEMIN IS .FALSE.
C IF STRNG IS EMPTY SEMIN IS .FALSE.
      CHARACTER*80 SSTRNG
      CHARACTER* (*) RSTRNG
      SEMIN=.FALSE.
      IF (SSTRNG(1:2) .EQ. '  ') RETURN
C NO NEED TO LOOK
      ILAST=1
C COUNTER FOR POSITION IN SSTRNG
   20 ICHAR=INDEX(SSTRNG(ILAST:80),';')
C LOOK FOR ; AS TERMINATOR OF STRING
      IF (ICHAR .EQ. 0) GOTO 25
C HAVENT FOUND A MATCH BY THE END OF THE LINE
      IF (INDEX(RSTRNG,SSTRNG(ILAST:ILAST+ICHAR-2)) .NE. 0) RETURN
C LOOK BETWEEN ;S IN SSTRNG AND IF FIND A MATCH RETURN
      ILAST=ILAST+ICHAR
C POSITION AFTER THE ;
      IF (ILAST .LE. 79) GOTO 20
C BACK ROUND IF NOT AT END OF LINE
C
C ELSE END
   25 CONTINUE
```

```
      SEMIN=.TRUE.
C NO MATCH
      RETURN
      END
$TITL  FSTJDN.FTN - POINT IN POLYGON
C PROGRAM COLLECTION FASTJORDAN
C    SALOMON,K.B., 1978. AN EFFICIENT POINT-IN-POLYGON ALGORITHM
C      COMPUTERS AND GEOSCIENCES, V4, NO. 2, P. 173-178
C
C USE BY READING VERTICES OF POLYGON IN ORDER (BUT NOT CLOSING IT)
C AND CALLING PREPLY ONCE BEFORE THE SEARCH IS BEGUN.  THE ACTUAL
C SEARCHING IS DONE BY MEANS OF THE LOGICAL FUNCTION INOUT.
C
      SUBROUTINE PREPLY(X,Y,NUVERT,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*****************************************************************************************
C  THIS ROUTINE PREPARES THE POLYGON CONSISTING OF THE NUVERT VERTICES
C  (X(I),Y(I)) BY FIRST SORTING THE SEGMENT Y-END POINTS INTO
C  DECREASING ORDER AND FORMING AN INTERVAL FOR EACH CONSECUTIVE PAIR:
C  (YINTVL(I),YINTVL(I+1)), I=1,INTVLS.  THIS IS PERFORMED BY CALLING
C  SORT.
C       THE CODE CONSISTING OF THE DO 100 AND DO 200 LOOPS CONSTRUCTS,
C  FOR EACH INTERVAL I, THE LIST OF SEGMENTS TO BE TESTED BY INOUT.
C  THIS LIST IS PLACED IN THE I-TH ROW OF SWATH.  THE FIRST ENTRY,
C  SWATH(I,1), WILL BE SET TO THE NUMBER OF SEGMENTS IN THE ROW.  NOTE
C  THAT AS YINTVL CONTAINS NO REDUNDANCIES, I.E.  YINTVL(I) IS STRICTLY
C  GREATER THAN YINTVL(I+1), NO HORIZONTAL SEGMENTS WILL BE PLACED IN
C  THE LIST.
C       THE CODE CONSISTING OF THE DO 300 LOOP ESTABLISHES THE
C  RECIPROCAL SLOPE FOR EACH NON-HORIZONTAL SEGMENT.  THIS IS TO BE
C  USED BY INOUT.  FINALLY, THE SEGMENTS WITHIN A ROW OF SWATH ARE
C  OREDERED FROM LEFT-TO-RIGHT.
C
C*****************************************************************************************
C
      INTEGER*2 SWATH(100,25)
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100)
      CALL SORT(Y,NUVERT,YINTVL,INTVLS)
      IF (INTVLS .LE. 0) GOTO 400
      X(NUVERT+1)=X(1)
      Y(NUVERT+1)=Y(1)
      DO 100 I=1,INTVLS
  100 SWATH(I,1)=0
      DO 200 I=1,INTVLS
         DO 200 J=1,NUVERT
         IF (Y(J).GE.YINTVL(I) .AND. YINTVL(I+1).GE.Y(J+1) .OR.
     *        Y(J+1).GE.YINTVL(I) .AND. YINTVL(I+1).GE.Y(J))
     *                                             CALL INCLUD(SWATH,I,J)
  200 CONTINUE
      DO 300 I=1,NUVERT
      IF (Y(I).EQ.Y(I+1)) GOTO 300
      RSLOPE(I)=(X(I+1)-X(I))/(Y(I+1)-Y(I))
  300 CONTINUE
      CALL ORDER(X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
      RETURN
  400 WRITE(7,401)
  401 FORMAT(' ******* PREP OF POLYGON ABORTED SINCE NO INTERVALS',
     * ' CONSTRUCTED')
      STOP
      END
      SUBROUTINE SORT(Y,NUVERT,YINTVL,INTVLS)
C
C*************************************************************************************
C
C  ROUTINE ESTABLISHES THE INTERVALS OF THE Y-AXIS DEFINED BY THE
C  ENDPOINTS OF THE POLYGON'S SEGMENTS.  THE DO 100 LOOP INITIALLISES
C  YSORT FROM THE SEGMENT Y-END POINTS.  THE DO 200 LOOPS SORT YSORT
C INTO DESCENDING ORDER.  THE DO 300 LOOP ELIMINATES REDUNDANCIES IN
C  YSORT AND PLACES IRREDUNDANT SORTED Y'S INTO YINTVL.  IT ALSO SETS
C  INTVLS TO THE TRUE NUMBER OF Y INTERVALS.  JUST PRIOR TO RETURNING
C  A FINAL INTERVAL EXTENDING TO '-INFINITY' IS ESTABLISHED.
C
C*************************************************************************************
C
      REAL Y(100),YINTVL(100),YSORT(100)
      INTEGER*2 UPPER
      DO 100 I=1,NUVERT
  100 YSORT(I)=Y(I)
      UPPER=NUVERT-1
      DO 200 I=1,UPPER
         IPLS1=I+1
         DO 200 J=IPLS1,NUVERT
         IF (YSORT(I).GE.YSORT(J)) GOTO 200
             TEMP=YSORT(I)
             YSORT(I)=YSORT(J)
             YSORT(J)=TEMP
  200 CONTINUE
      YINTVL(1)=YSORT(1)
      INTVLS=0
      DO 300 I=1,UPPER
         IF (YSORT(I).EQ.YSORT(I+1)) GOTO 300
         INTVLS=INTVLS+1
         YINTVL(INTVLS+1)=YSORT(I+1)
  300 CONTINUE
      YINTVL(INTVLS+2)=-1.0E75
      RETURN
      END
      SUBROUTINE INCLUD(SWATH,I,J)
C
C*************************************************************************************
C  ROUTINE PLACES THE J-TH POLYGON SEGMENT INTO THE NEXT AVAILABLE
C  LOCATION IN ROW I OF SWATH.
C
C*************************************************************************************
C
      INTEGER*2 SWATH(100,25),POINTR
      SWATH(I,1)=SWATH(I,1)+1
      POINTR=SWATH(I,1)
      SWATH(I,POINTR+1)=J
      RETURN
      END
      SUBROUTINE ORDER(X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*************************************************************************************
C
C  FOR EACH INTERVAL, A HORIZONTAL LINE IS PASSED THROUGH THE MIDDLE
C  (YMID) OF THE INTERVAL.  THE DO 100 LOOP PLACES THE X-INTERSECTION
C  OF EACH SEGMENT IN THIS SWATH SO THAT THESE INTERSECTIONS OCCUR
C  FROM LEFT-TO-RIGHT.
C
```

```
C*****************************************************************************************
C
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100),XINTSC(25)
      INTEGER*2 SWATH(100,25),POINTR,SEGNO,UPPER
      LOGICAL VERTSG
      DO 200 INTVAL=1,INTVLS
         NMBSEG=SWATH(INTVAL,1)
         YMID=(YINTVL(INTVAL)+YINTVL(INTVAL+1))/2.0
         DO 100 POINTR=1,NMBSEG
            SEGNO=SWATH(INTVAL,POINTR+1)
            VERTSG=ABS(X(SEGNO+1)-X(SEGNO)) .LT.  1.0E-5
            IF (VERTSG) XINTSC(POINTR)=X(SEGNO)
            IF (.NOT.  VERTSG) XINTSC(POINTR)=X(SEGNO)+
     *                                   RSLOPE(SEGNO)*(YMID-Y(SEGNO))
  100    CONTINUE
         IF (NMBSEG.LT.2 .OR.  MOD(NMBSEG,2).NE.0) GOTO 300
         UPPER=NMBSEG-1
         DO 200 I=1,UPPER
            IPLS1=I+1
            DO 200 J=IPLS1,NMBSEG
               IF (XINTSC(I).LE.XINTSC(J)) GOTO 200
               TEMP=XINTSC(I)
               XINTSC(I)=XINTSC(J)
               XINTSC(J)=TEMP
               ITEMP=SWATH(INTVAL,I+1)
               SWATH(INTVAL,I+1)=SWATH(INTVAL,J+1)
               SWATH(INTVAL,J+1)=ITEMP
  200 CONTINUE
      RETURN
  300 WRITE(7,301) INTVAL
  301 FORMAT(' ** PREP OF POLYGON ABORTED.  INTERVAL ',I5/
     * ' HAS EITHER LESS THAN TWO SEGMENTS OR AN ODD NUMBER OF THEM')
      STOP
      END
      LOGICAL FUNCTION INOUT(XP,YP,X,Y,YINTVL,INTVLS,SWATH,RSLOPE)
C
C*******************************************************************************
C
C  THE FOUR LINES ENCLOSED IN DASHES DETERMINE THE INTERVAL CONTAINING
C  YP.  THE DO 400 LOOP CONTINUES UNTIL THE FIRST SEGMENT WITHIN THE
C  INTERVAL FALL TO THE LEFT OF (XP,YP).  IN THIS EVENT, INOUT IS SET
C  .TRUE.  IFF AN EVEN NUMBER OF SEGMENTS HAS BEEN TESTED.
C
C*****************************************************************************************
C
      REAL X(100),Y(100),YINTVL(100),RSLOPE(100)
      INTEGER*2 SWATH(100,25),SEGNO
      INOUT=.FALSE.
C-----------------------------------------------------
      INTVAL=0
  100 INTVAL=INTVAL+1
      IF (YINTVL(INTVAL) .GT.  YP) GOTO 100
      INTVAL=INTVAL-1
C-----------------------------------------------------
  300 IF (INTVAL.LT.1 .OR.  INTVAL.GT.INTVLS) RETURN
      NMBSEG=SWATH(INTVAL,1)+1
      DO 400 I=2,NMBSEG
         SEGNO=SWATH(INTVAL,I)
         IF(XP-X(SEGNO).LE.(YP-Y(SEGNO))*RSLOPE(SEGNO)) GOTO 500
  400 CONTINUE
```

```
      RETURN
  500 INOUT=MOD(I,2) .EQ. 1
      RETURN
      END
```