UR1983_48

1/24

# 1983/48. FORTRAN programs for the implementation of MIRLOCH

*R.G. Richardson*

*Abstract*

The program suite described is used for adding entries to the Mineral Resources Location And Characterisation System (MIRLOCH) of the Economic Geology Section, sorting the data-base, and searching it. The data are entered in a simple fill-in the spaces manner and part or all of the details entered may be searched.

## 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 MIRADD and continues until END is typed in response to the reference number prompt. Data are entered left justified in response to prompts, with the host, form, and exploration numbers being entered as one or more digits on the same line.

At the end of input a proof-sheet is output and the data saved in file MIRTEMP.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 MIRTEMP.TMP must be added to the end of the existing file by typing MIRMERGE. To ensure that the file remains in reference number order, it should be sorted periodically using the MIRSORT command. As sorting is a slow process it should not be carried out more often than necessary.

To allow for changes in the status of a mineral resource the main data file may be altered using the MIREDIT command. The editing should be terminated using the DONE command. Care should be taken to ensure that there is no reformatting of the data records.

### *Data-base searching*

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

The program requests the following data:

(i) Quadrangle number - one or more pairs of digits on a single line, each pair terminated by /. Each pair represents a geological atlas quadrangle number. If a search by quadrangle is not required leave the line blank.

*e.g.* 24/32/77/ Quadrangles 24, 32 or 77

(ii) Mine/deposit name - one or more names on a single line, each name terminated by /. If a search by name is not required leave the line blank.

*e.g.* COMET/LYELL/ABERFOYLE/

(iii) Commodity - details as for mine/deposit name.

*e.g.* CU/SN/FE/ Copper, tin, or iron.

(iv) Search by area - N for no search by area
- Y for finding entries inside a specified co-ordinated area.
Entries on the boundary of the area will not be found.
The program prompts for the number of vertices and the vertex co-ordinates in the same form as used on the input sheets.

(v) Map sheets - a series of sheet numbers on a single line, each terminated by /.

*e.g.* 83161/82143/

(vi) Status - one or more status numbers on a single line, each terminated by /. If a search by status is not required leave the line blank.

*e.g.* 0/2/ Operating mine or non-operating mine with unknown reserves.

(vii) Size of deposit - as for status.

*e.g.* 1/4/ Very small or large.

(viii) Host rock - as for status.

*e.g.* 0/6/ Precambrian sequences or Mathinna Beds.

(ix) Age of mineralisation - as for status.

*e.g.* 0/1/ Not determined or Precambrian.

(x) Form of deposit - as for status.

| | | |
|---|---|---|
| *e.g.* | 0/3/ | Not determined or stockwork. |
| | 23/45/ | Vein and stockwork or disseminated and massive. |

(xi) Exploration of deposit - as for status.

*e.g.* 2/3/ Geological mapping or geochemical surveys.

(xii) Printout required - Y if the number of entries found can be realistically printed.
N if the printout is not wanted.

## THE PROGRAMS

### *MIRADD (Appendix 2)*

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

*MIRMERGE (Appendix 3)*

The data from MIRTEMP.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.

*MIRSORT (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.

*MIRSRCH (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.

*MIREDIT (Appendix 6)*

Allows the user to edit the data-base (MIRLOCH.DAT) using the standard editing facilities.

## DATA FORMAT

The data-base (MIRLOCH.DAT) has the following arrangement of data:

| *Starting column* | *Contents of field* |
|---|---|
| 1 | Reference number |
| 6 | Mine/deposit name |
| 36 | Commodity |
| 51 | A.M.G. Co-ords. |
| 62 | Map sheet |
| 67 | Status |
| 68 | Size of deposit |
| 69 | Host rock |
| 71 | Age of mineralisation |
| 72 | Form of deposit |
| 74 | Exploration of deposit |

[21 September 1983]

## APPENDIX 1

Data format sheet

# MINERAL RESOURCES LOCATION AND CHARACTERISATION SYSTEM

(Data File : MIRLOCH)

Reference No. _ _ _ _ _ (5) [First two numbers refer to geological atlas quadrangle]

Mine/deposit name _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ (30)

Commodity(s) _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ (15) [List chemical symbols for metallic resources]

A.M.G. Coords _ _ _ _ _ _ _ _ _ _ _ (11) [m E (5 figures) m N (6 figures)]

Map sheet _ _ _ _ _ (5) [1:50 000 National Map index sheet numbers]

Status (1)

| | | | | | |
|---|---|---|---|---|---|
| 0 | Operating mine | 3 | Abandoned mine - reserves known | 6 | Prospect - explored |
| 1 | Non-operating mine - reserves known | 4 | Abandoned mine - reserves unknown | 7 | Prospect - unexplored |
| 2 | Non-operating mine - reserves unknown | 5 | Abandoned - mined out | 8 | Mineralized area |
| | | | | 9 | Mineral occurrence |

Size of deposit (1) [tonnes or $m^3$]

| | | | | | |
|---|---|---|---|---|---|
| 0 | Not determined | 2 | Small ($10^2$ - $10^4$ t) | 4 | Large ($10^6$ - $10^7$ t) |
| 1 | Very small (< $10^2$ t) | 3 | Medium ($10^4$ - $10^6$ t) | 5 | Very large (> $10^7$ t) |

Host rock (2)

| | | | | | |
|---|---|---|---|---|---|
| 0 | Precambrian sequences | 4 | Owen Cong/Moina Sst and correl. | 7 | Devonian granite |
| 1 | Cambrian sedimentary sequences | 5 | Gordon Lst/Eldon Gp and correl. | 8 | Parmeener Super Gp |
| 2 | Cambrian igneous sequences | 6 | Mathinna Beds | 9 | Jur.-Cen. sequences |
| 3 | Mount Read Volcanics and correl. | | | | |

Age of mineralization (1)

| | | | | | |
|---|---|---|---|---|---|
| 0 | Not determined | 4 | Ordov.-E. Devonian | 7 | Jurassic-Cretac. |
| 1 | Precambrian | 5 | L. Devonian (granite) | 8 | Tertiary |
| 2 | Eo.-E. Cambrian | 6 | Permo-Triassic | 9 | Quaternary |
| 3 | M.-L. Cambrian | | | | |

Form of deposit (2)

| | | | | | |
|---|---|---|---|---|---|
| 0 | Not determined | 4 | Disseminated | 7 | Placer |
| 1 | Stratabound-stratiform | 5 | Massive | 8 | Residual |
| 2 | Vein (single, sheet, saddle) | 6 | Pipe/Lens | 9 | Other |
| 3 | Stockwork | | | | |

Exploration of deposit (5)

| | | | | | |
|---|---|---|---|---|---|
| 0 | Nil | 2 | Geological mapping | 4 | Geophysical surveys |
| 1 | Prospecting | 3 | Geochemical surveys | 5 | Drilling |

48-5

## APPENDIX 2

Program MIRADD

```
*MIRADD.CSS
*FOR RUNNING MIRADD AND PRINTING A PROOF SHEET
$IFX MIRTEMP.TMP; $WRITE ** CHECK LAST MIRMERGE; $EXIT; $ENDC
PRE ETM; AL MIRTEMP.TMP,IN,89
* SET UP THE OUTPUT FILE
L MIRADD,3; AS 6,MIRTEMP.TMP; * LOAD PROG AND SET LU 6
TEMPFILE 3,IN,80; ST
PRINT MIRTEMP.TMP
$WRITE  FILE IS MIRTEMP.TMP; ENA ETM; $EXIT
```

```
$TITL  MIRADD.FTN  -  ADD PROSPECTS TO THE MIRLOCH FILE
C TAKES KEYBOARD INPUT AND PREPARES A PROOF SHEET
      CHARACTER*30 NAME
      CHARACTER*15 COMMOD
      CHARACTER*11 MAPREF
      CHARACTER*5 REFNO,SHEET,EXPLOR
      CHARACTER*1 STATUS,SIZE,AGE
      CHARACTER*2 HOST,FORM
C INPUT IS REFERENCE NO.,MINE/DEPOSIT NAME,COMMODITIES,
C           MAP REFERENCE,MAP SHEET,STATUS,SIZE OF DEPOSIT,
C           HOST ROCK (UP TO 2),AGE OF MINERALISATION,
C           FORM OF DEPOSIT (UP TO 2),EXPLORATION OF DEPOSIT
C           (UP TO 5)
C
      OPEN(UNIT=5,FILE='CON:')
C SET THE CONSOLE UP AS LU 5 FOR INPUT
C THE OUTPUT IS TO LU 6
   10 WRITE(5,100)
  100 FORMAT(' REF NO. OR END')
      READ(5,200) REFNO
  200 FORMAT(A5)
      IF (INDEX(REFNO,'END ') .NE. 0) GOTO 20
C RECOGNISE END BY HAVING END ON ITS OWN AND PRINT PROOF SHEET
      WRITE(5,101)
  101 FORMAT(' NAME')
      READ(5,201) NAME
  201 FORMAT(A30)
      WRITE(5,102)
  102 FORMAT(' COMMODITY')
      READ(5,202) COMMOD
  202 FORMAT(A15)
      WRITE(5,103)
  103 FORMAT(' COORDS')
      READ(5,203) MAPREF
  203 FORMAT(A11)
      WRITE(5,104)
  104 FORMAT(' MAP SHEET')
      READ(5,204) SHEET
  204 FORMAT(A5)
      WRITE(5,105)
  105 FORMAT(' STATUS')
      READ(5,205) STATUS
  205 FORMAT(A1)
      WRITE(5,106)
  106 FORMAT(' SIZE')
      READ(5,205) SIZE
      WRITE(5,107)
  107 FORMAT(' HOST ROCK')
      READ(5,206) HOST
  206 FORMAT(A2)
      WRITE(5,108)
  108 FORMAT(' AGE')
      READ(5,205) AGE
      WRITE(5,109)
  109 FORMAT(' FORM')
      READ(5,206) FORM
      WRITE(5,110)
  110 FORMAT(' EXPLORATION')
```

```
      READ(5,204) EXPLOR
C READ ONE COMPLETE DATA SHEET
C
C SO WRITE TO OUTPUT FILE
      WRITE(6,300) REFNO,NAME,COMMOD,MAPREF,SHEET,STATUS,SIZE,
     .             HOST,AGE,FORM,EXPLOR
  300 FORMAT(A5,'@',A30,'@',A15,'@',A11,'@',A5,'@',A1,'@',A1,
     .       '@',A2,'@',A1,'@',A2,'@',A5,'@')
C WRITE WITH DELIMITERS AFTER EACH FIELD
      GOTO 10
C BACK AROUND
C
C OTHERWISE CLOSE UP AND STOP
   20 CONTINUE
      CLOSE(UNIT=6,STATUS='KEEP')
      END
```

## APPENDIX 3

Program MIRMERGE

```
*MIRMERGE.CSS
* FOR MERGING CORRECTED FILE WITH MAIN FILE
PRE ETM; XDE MRGTMP.TMP; *DELETE SCRATCH FILE
AL MRGTMP.TMP,IN,78; L MIRMERGE,10; * AL SCRATCH FILE AND LOAD PROG
REP MIRTEMP.TMP,FF00; AS 6,MIRTEMP.TMP,ERO; AS 4,MRGTMP.TMP; ST
$IFNE 0; $WRITE MIR TRANSLATE ERROR; ENA ETM;  $EXIT; $ENDC
REP MRGTMP.TMP,FF00
$BUILD COPY.CMD
IN MIRLOCH.DAT
AL TEMP:TEMP.DAT,IN,78/6/5
OUT TEMP:TEMP.DAT
COPY *,*
IN MRGTMP.TMP
COPY *,*
END
$ENDB
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE MIR COPY-MERGE ERROR; ENA ETM; $EXIT; $ENDC
REP MIRLOCH.DAT,0; REP TEMP:TEMP.DAT,FF00
DE COPY.CMD,MIRLOCH.DAT
$BUILD COPY.CMD
IN TEMP:TEMP.DAT
AL MIRLOCH.DAT,IN,78/10/3
OUT MIRLOCH.DAT
COPY *,*
REW I
REW O
VERIFY *,*
END
$ENDB
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE MIR COPY BACK FAILED; ENA ETM; $EXIT; $ENDC
REP MIRLOCH.DAT,FF00; REP TEMP:TEMP.DAT,0; REP MIRTEMP.TMP,0
REP MRGTMP.TMP,0
DE COPY.CMD,MRGTMP.TMP,MIRTEMP.TMP,TEMP:TEMP.DAT
ENA ETM
$EXIT
```

```
$TITL  MIRMERGE.FTN - CONVERT PROOF FILE TO MASTER FILE FORMAT
C
      CHARACTER*30 NAME
      CHARACTER*15 COMMOD
      CHARACTER*11 MAPREF
      CHARACTER*5 REFNO,SHEET,EXPLOR
      CHARACTER*1 STATUS,SIZE,AGE
      CHARACTER*2 HOST,FORM
C AS FOR MIRADD.FTN
C
C NOW SET UP EQUIVALENCE
      CHARACTER*1 BUFF(78)
      EQUIVALENCE (BUFF(1),REFNO),(BUFF(6),NAME),(BUFF(36),COMMOD),
     . (BUFF(51),MAPREF),(BUFF(62),SHEET),(BUFF(67),STATUS),
     . (BUFF(68),SIZE),(BUFF(69),HOST),(BUFF(71),AGE),
     . (BUFF(72),FORM),(BUFF(74),EXPLOR)
C THIS IS USED TO ALLOW BINARY OUTPUT FOR FASTER ACCESS
C
      OPEN(UNIT=4,FORM='BINARY',RECL=78)
C OPEN THE OUTPUT FILE FOR BINARY OUTPUT
   10 READ(6,300,END=20) REFNO,NAME,COMMOD,MAPREF,SHEET,STATUS,SIZE,
     .                HOST,AGE,FORM,EXPLOR
  300 FORMAT(A5,1X,A30,1X,A15,1X,A11,1X,A5,1X,A1,1X,A1,
     .       1X,A2,1X,A1,1X,A2,1X,A5)
      WRITE(4) BUFF
C BINARY WRITE TO OUTPUT FILE
      GOTO 10
C BACK ROUND
C
C AT END
   20 CONTINUE
      CLOSE(UNIT=4,STATUS='KEEP')
      CLOSE(UNIT=6,STATUS='KEEP')
      END
```

APPENDIX 4

Program MIRSORT

```
*MIRSORT.CSS
* FOR SORTING MIRLOCH FILE INTO CHRONOLOGICAL ORDER
PRE ETM
L MIRSORT,10; AS 4,MIRLOCH.DAT,ERO; AL TEMP:TEMP.DAT,IN,88/5/2
AS 6,TEMP:TEMP.DAT; ST
$IFNE 0; $WRITE MIR SORT ERROR; ENA ETM; $EXIT; $ENDC
$BUILD COPY.CMD
IN TEMP:TEMP.DAT
AL MIRLOCH.DAT,IN,88/10/5
OUT MIRLOCH.DAT
COPY *,*
REW I
REW O
VERIFY *,*
END
$ENDB
REP TEMP:TEMP.DAT,FF00; REP MIRLOCH.DAT,0; DE MIRLOCH.DAT
L COPY32,50; ST ,COM=COPY.CMD,LI=NULL:,LO=NULL:
$IFNE 0; $WRITE MIR SORT-COPY ERROR; ENA ETM; $EXIT; $ENDC
REP MIRLOCH.DAT,FF00; REP TEMP:TEMP.DAT,0; DE TEMP:TEMP.DAT,COPY.CMD
ENA ETM; $EXIT
```

```
$TITL MIRSORT.FTN - SORT MIRLOCH FILE INTO QUAD AND NUMBER ORDER
C
      CHARACTER*30 NAME
      CHARACTER*15 COMMOD
      CHARACTER*11 MAPREF
      CHARACTER*5 REFNO,SHEET,EXPLOR
      CHARACTER*1 STATUS,SIZE,AGE
      CHARACTER*2 HOST,FORM
C AS FOR MIRADD.FTN
C
C NOW SET UP EQUIVALENCE
      CHARACTER*1 BUFF(78)
      EQUIVALENCE (BUFF(1),REFNO),(BUFF(6),NAME),(BUFF(36),COMMOD),
     . (BUFF(51),MAPREF),(BUFF(62),SHEET),(BUFF(67),STATUS),
     . (BUFF(68),SIZE),(BUFF(69),HOST),(BUFF(71),AGE),
     . (BUFF(72),FORM),(BUFF(74),EXPLOR)
C THIS IS USED TO ALLOW BINARY OUTPUT FOR FASTER ACCESS
C
C FOLLOWING ARE FOR THE SORT
      INTEGER*2 INDEX(4000)
      INTEGER*4 IA(4000)
C ALLOW FOR 4000 ENTRIES
C
C
      OPEN(UNIT=4,FORM='BINARY',ACCESS='DIRECT',RECL=78)
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. 4000) STOP 'MORE THAN 4000 ENTRIES'
      INDEX(IREC)=IREC
C FILL SORT INDEX
      IA(IREC)=CTOI(REFNO,K)
C GET THE REFERENCE NUMBER
      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=78)
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*4 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 MIRSRCH

```
*MIRSRCH.CSS - SEARCH THE MIRLOCH FILE
L MIRSRCH,10; AS 4,MIRLOCH.DAT,ERO; XAL TEMP:MIRLOCH.TMP,IN,132/3/2
AS 6,TEMP:MIRLOCH.TMP; REW 6; AS 5,CON:; ST
$IFX TEMP:MIRLOCH.TMP; PRI TEMP:MIRLOCH.TMP,DEL; $ENDC; $EXIT
```

```
$TITL  MIRSRCH.FTN - SEARCH MIRLOCH FILE
C
      CHARACTER*30 NAME
      CHARACTER*15 COMMOD
      CHARACTER*11 MAPREF
      CHARACTER*5 REFNO,SHEET,EXPLOR
      CHARACTER*1 STATUS,SIZE,AGE
      CHARACTER*2 HOST,FORM
C AS FOR MIRADD.FTN
C
C NOW SET UP EQUIVALENCE
      CHARACTER*1 BUFF(78)
      EQUIVALENCE (BUFF(1),REFNO),(BUFF(6),NAME),(BUFF(36),COMMOD),
     . (BUFF(51),MAPREF),(BUFF(62),SHEET),(BUFF(67),STATUS),
     . (BUFF(68),SIZE),(BUFF(69),HOST),(BUFF(71),AGE),
     . (BUFF(72),FORM),(BUFF(74),EXPLOR)
C THIS IS USED TO ALLOW BINARY OUTPUT FOR FASTER ACCESS
C
      CHARACTER*80 SQUAD,SNAME,SCOM,SMAP,SSTAT,SSIZE,SHOST,
     .  SAGE,SFORM,SXPLOR
      INTEGER*2 FFEED,STYPE
C USED TO PUT FORM FEEDS INTO OUTPUT FILE
      INTEGER*4 EAST,NORTH
C USED FOR MAP COORDS
      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
      INTEGER*4 FCOUNT
      REAL AHOST(2),AFORM(2),AXPLOR(5)
      REAL STATCN(10),SIZCON(6),HOSTCN(10),AGECON(10),
     .     FORMCN(10),XPLCON(6)
      DATA STATCN/'OPM','NOR','NOX','AMR','AMX','AMO',
     . 'PEX','PUN','MAR','MOC'/
C THE POSSIBLE STATUS ABBREVS
      DATA SIZCON/'ND','VS','SM','ME','LA','VL'/
C THE SIZE ABBREVS
      DATA HOSTCN/'PCS','CSS','CIG','MRV','OMS','GLE','MAT',
     .      'DGN','PSG','JCS'/
C THE HOST ROCK ABBREVS
      DATA AGECON/'ND','PC','EC','MC','OD','LD','PT','JC','TT','QT'/
C THE AGE ABBREVS
      DATA FORMCN/'  ND','STAT','VEIN','STWK','DISS','MASS','PIPE',
     .      'PLAC','RESD','OTHR'/
C THE FORM ABBREVS
      DATA XPLCON/'NO','PS','GM','GC','GP','DR'/
C THE EXPLORATION ABBREVS
      DATA FFEED/3072/
C THE FORM FEED CHARACTER IN A1
      OPEN(UNIT=4,FORM='BINARY',RECL=78)
C OPEN THE INPUT FILE
      WRITE(5,100)
  100 FORMAT(' SEARCHING JOB RECORDS')
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(' QUADRANGLE?')
      READ(5,201) SQUAD
  201 FORMAT(A80)
      IF (SQUAD(1:1) .NE. ' ') WRITE(6,401) SQUAD
  401 FORMAT(' QUADRANGLE',10X,A80)
      WRITE(5,103)
  103 FORMAT(' MINE/DEPOSIT NAME?')
      READ(5,201) SNAME
      IF (SNAME(1:1) .NE. ' ') WRITE (6,402) SNAME
  402 FORMAT(' MINE/DEPOSIT NAME',10X,A80)
      WRITE(5,104)
  104 FORMAT(' COMMODITY?')
      READ(5,201) SCOM
      IF (SCOM(1:1) .NE. ' ') WRITE(6,403) SCOM
  403 FORMAT(' COMMODITY',10X,A80)
   22 WRITE(5,108)
  108 FORMAT(' SEARCH FOR SITES WITHIN A SPECIFIED AREA (Y OR N)?')
      READ(5,200) STYPE
  200 FORMAT(A1)
      IF (STYPE .EQ. 'Y') GOTO 23
C WANT TO READ COORDS
      IF (STYPE .NE. 'N') GOTO 22
C INVALID OPTION
      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?')
      READ(5,*) NCOORD
      IF (NCOORD .LE. 2) GOTO 23
C NEED AT LEAST 3 VERTICES TO DEFINE AN AREA
      WRITE(5,110)
  110 FORMAT(' ENTER VERTEX COORDS IN CLOCKWISE ORDER'/
     . ' ONE PER LINE IN FORM EASTTNORTHH')
      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(I5,I6)
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*0.01
      Y(I)=IY*0.01
C CONVERT TO KM BEFORE STORING
   24 CONTINUE
      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
C
   26 CONTINUE
      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(5,111)
  111 FORMAT(' STATUS (0-9)?')
      READ(5,201) SSTAT
      IF(SSTAT(1:1) .NE. ' ') WRITE(6,409) SSTAT
  409 FORMAT(' STATUS',10X,A80)
      WRITE(5,112)
  112 FORMAT(' SIZE (0-5)?')
      READ(5,201) SSIZE
      IF (SSIZE(1:1) .NE. ' ') WRITE(6,410) SSIZE
  410 FORMAT(' DEPOSIT SIZE',10X,A80)
      WRITE(5,114)
  114 FORMAT(' HOST ROCK (0-9)?')
      READ(5,201) SHOST
      IF (SHOST(1:1) .NE. ' ') WRITE(6,411) SHOST
  411 FORMAT(' HOST ROCK',10X,A80)
      WRITE(5,115)
  115 FORMAT(' AGE (0-9)?')
      READ(5,201) SAGE
      IF (SAGE(1:1) .NE. ' ') WRITE(6,412) SAGE
  412 FORMAT(' AGE OF MIN',10X,A80)
      WRITE(5,116)
  116 FORMAT(' FORM OF DEPOSIT (0-9)?')
      READ(5,201) SFORM
      IF (SFORM(1:1) .NE. ' ') WRITE(6,413) SFORM
  413 FORMAT(' FORM OF DEPOSIT',10X,A80)
      WRITE(5,117)
  117 FORMAT(' EXPLORATION (0-5)?')
      READ(5,201) SXPLOR
      IF (SXPLOR(1:1) .NE. ' ') WRITE(6,414) SXPLOR
  414 FORMAT(' EXPLORATION',10X,A80)
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(SCOM,COMMOD)) GOTO 27
```

```
C NOT THE RIGHT COMMODITY
      IF(NOTIN(SHOST,HOST)) GOTO 27
C NOT THE HOST ROCK
      IF (NCOORD .NE. 0) THEN
       EAST=CTOI(MAPREF(1:5),K)
       NORTH=CTOI(MAPREF(6:11),K)
C UNPACK FROM MAPREF
       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
       XTMP=EAST*0.01
       YTMP=NORTH*0.01
C CONVERT TO KM
       IF (.NOT. INOUT(XTMP,YTMP,X,Y,YINTVL,INTVLS,SWATH,RSLOPE))
     . GOTO 27
C NOT IN THE SPECIFIED AREA
      ENDIF
      IF (SAGE(1:1) .NE. ' ' .AND. INDEX(SAGE,AGE) .EQ. 0) GOTO 27
C NOT THE DESIRED AGE
      IF (NOTIN(SFORM,FORM)) GOTO 27
C NOT THE FORM OF DEPOSIT
      IF (SSIZE(1:1) .NE. ' ' .AND. INDEX(SSIZE,SIZE) .EQ. 0) GOTO 27
C NOT THE RIGHT SIZE
      IF (NOTIN(SQUAD,REFNO(1:2))) GOTO 27
C NOT RIGHT QUADRANGLE
      IF (NOTIN(SNAME,NAME)) GOTO 27
C NOT RIGHT NAME
      IF (NOTIN(SXPLOR,EXPLOR)) GOTO 27
C NOT RIGHT FORM OF EXPLORATION
      IF (SSTAT(1:1) .NE. ' ' .AND. INDEX(SSTAT,STATUS) .EQ. 0)
     . GOTO 27
C NOT RIGHT STATUS
C IF FINALLY GET TO HERE HAVE FOUND AN ENTRY THAT MATCHES THE
C SEARCH OPTIONS
C
      IF (MOD(FCOUNT,56) .EQ. 0) CALL HEAD
C HEAD PAGE
      FCOUNT=FCOUNT+1
C INCREMENT FOUND ENTRY COUNTER
C
C NOW SUBSTITUTE MNEMONICS FOR THE CODE NUMBERS BEFORE PRINTING
      CALL NMONIC(STATUS,ASTAT,STATCN,10)
      CALL NMONIC(SIZE,ASIZE,SIZCON,6)
      CALL NMONIC(HOST(1:1),AHOST(1),HOSTCN,10)
      CALL NMONIC(HOST(2:2),AHOST(2),HOSTCN,10)
      CALL NMONIC(AGE,AAGE,AGECON,10)
      CALL NMONIC(FORM(1:1),AFORM(1),FORMCN,10)
      CALL NMONIC(FORM(2:2),AFORM(2),FORMCN,10)
      DO 33 I=1,5
      CALL NMONIC(EXPLOR(I:I),AXPLOR(I),XPLCON,6)
   33 CONTINUE
      WRITE(6,416) REFNO,NAME,COMMOD,MAPREF,SHEET,ASTAT,ASIZE,
     .            AHOST,AAGE,AFORM,AXPLOR
  416 FORMAT(A5,2X,A30,2X,A15,2X,A11,2X,A5,4X,A3,4X,A2,4X,A3,
     .      ',',A3,3X,A2,3X,A4,',',A4,2X,4(A2,','),A2)
      GOTO 27
C BACK AROUND
C
C NOW AT THE END OF THE RUN
   40 CONTINUE
```

```
      WRITE(5,118) FCOUNT
  118 FORMAT(I7,' RECORDS WERE FOUND'/' DO YOU WANT A',
     .         ' PRINTOUT (Y OR N)?')
      READ(5,200) STYPE
      IF (STYPE .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='MIRNMON.TXT',SHARE='ERO')
C OPEN THE FILE OF EXPANSIONS OF ABBREVIATIONS
      WRITE(6,418) FFEED
  418 FORMAT(A1)
   45 READ(7,419,END=46) BUFF
  419 FORMAT(78A1)
      WRITE(6,419) BUFF
      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//'REFNO',8X,'MINE/DEPOSIT NAME',10X,'COMMODITY(S)',
     . 4X,'AMG COORDS',3X,'SHEET',2X,'STATUS',2X,'SIZE',4X,
     . 'HOST',5X,'AGE',4X,'FORM',7X,'EXPLORATION')
      RETURN
      END
      SUBROUTINE NMONIC(CVAR,STRVAR,CONSTS,NOPT)
      CHARACTER*1 CVAR
      REAL STRVAR,CONSTS(2),BLANK
C USED TO PUT A STRING CONSTANT FROM CONSTS INTO STRVAR
C DEPENDING ON THE CHARACTER IN CVAR
C IF CVAR IS BLANK STRVAR IS SET BLANK
C NOPT IS THE NUMBER OF VALID OPTIONS FOR CVAR
      DATA BLANK/4H    /
      JVAL=CTOI(CVAR,K)+1
C CHARACTER TO INTEGER CONVERSION TO GET THE VALUE
      IF (CVAR .NE. ' ' .AND. JVAL .LE. NOPT) THEN
      STRVAR=CONSTS(JVAL)
      ELSE
      STRVAR=BLANK
      ENDIF
      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
```

## APPENDIX 6

Program MIREDIT

```
*MIREDIT.CSS - FOR EDITING MIRLOCH.DAT
REP MIRLOCH.DAT,0
G MIRLOCH.DAT
REP MIRLOCH.DAT,FF00
$EXIT
```