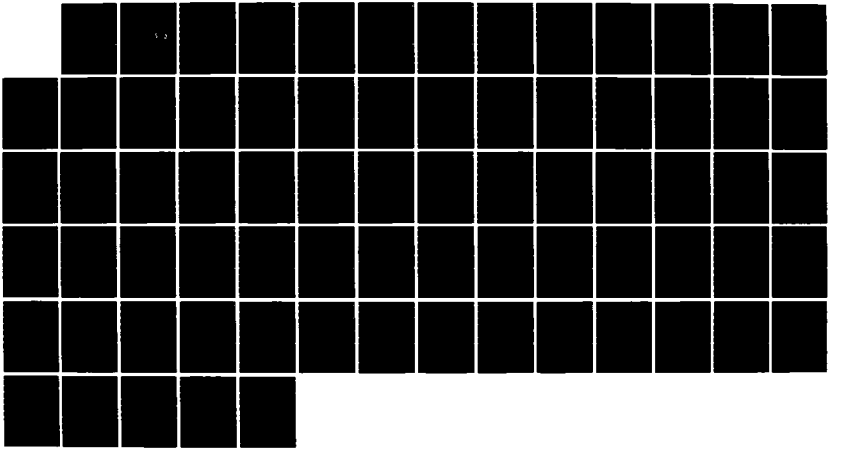
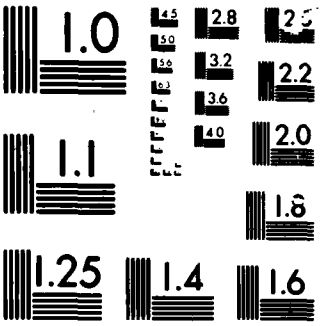
AD-A167 048 A DATA BASE EDITOR FOR MOODS (MASTER OCEANOGRAPHIC OBSERVATIONS DATA SET)(U) NAVAL OCEAN RESEARCH AND DEVELOPMENT ACTIVITY NSTL STATION MS W J TEAGUE ET AL. 1/1

UNCLASSIFIED FEB 86 NORDA-136 F/G 8/10 NL




1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY
CHART

Naval Ocean Research and Development Activity
February 1986 **Report 136**

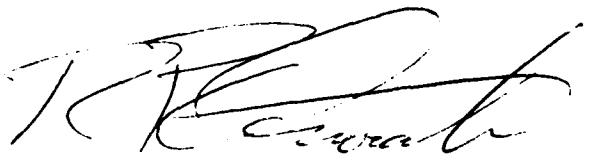
A Data Base Editor for MOODS

William J. Teague
Robert L. Pickett
Donald A. Burns
Oceanography Division
Ocean Science Directorate

Foreword

One of the most complete sets of oceanographic profile data available to the U.S. Navy is contained in the Master Oceanographic Observation Data Set (MOODS). MOODS is global and contains such physical oceanographic data as sea-surface temperature, bathythermographs, and salinity-temperature-depth records.

The primary requirements moving the Navy into MOODS development were to make climatologies available to the fleet and to provide environmental inputs to acoustical models. Bad data, naturally, blocks these goals and this report deals with a data base editor that attempts to ferret out erroneous data.



R. P. Onorati, Captain, USN
Commanding Officer, NORDA

Executive summary

The Master Oceanographic Observations Data Set (MOODS) contains 3.5 million data observations. The data, which include temperature and salinity profiles, are edited during updates, but this editing has been very superficial and allows for erroneous values. This editor attempts to ferret out the bad data by checking for oceanographic observations that are over land, above the sea surface, below the sea bottom; that have nonmonotonic, duplicate, or negative depths; that contain impossible or all-zero temperatures or salinities; that produce temperature or density inversions; that are misplaced either by location or by season; or that are duplicates. For four MOODS test sets (two Atlantic and two Pacific), the total rejection rate ranged 17–39%. Of these rejections, 9–16% were already flagged during update editing, 1–8% were rejected because of inversions and wild values, and 7–17% were duplicate and misplaced profiles.

Acknowledgments

This work was supported by the Nonacoustic ASW Oceanography Program, Program Element 63704N, through the Applied Oceanography and Geophysics Division, Kenneth M. Ferer, program manager.

Contents

A data base editor for MOODS

Introduction

Large oceanographic data bases are valuable for many applications: climatological estimates of surface and subsurface temperatures, heat content, layer depths, currents (via geostrophy), and frontal locations. Such estimates are useful for describing and comparing ocean regions, priming numerical models, and editing or calculating anomalies from present observations.

One of the most complete sets of oceanographic profile data is contained in the Master Oceanographic Observation Data Set (MOODS). MOODS is global, and contains such physical oceanographic data as sea-surface temperatures, mechanical and expendable (air and ship) bathythermographs, and hydrocast and other salinity-temperature-depth records. Source data were supplied by Argentina, Australia, England, France, Japan, Korea, Norway, and the United States (Fleet Numerical Oceanography Center, Lamont-Doherty Geophysical Laboratory, National Marine Fisheries Service, National Oceanographic Data Center, and Scripps Institution of Oceanography). These data (about 3.5 million observations for the years 1920–1982) are arranged by months and regions, have the same units (degrees centigrade, parts per thousand, meters), and are in a single format with sufficient information to trace sources (Bauer, 1985). The time interval between the time an observation is made and the inclusion of the data into MOODS is at least one year and is often several years.

The primary requirements driving the U. S. Navy into MOODS development were to make climatologies available to operational forces and to provide environmental inputs to acoustical models. MOODS brings together data, often acquired through different types of instrumentation, from many organizations. The quality of the data is often unknown. The characteristics of a particular data set can bias the data statistics in systematic ways, which are difficult to recognize unless there are observations to provide a context. Judgments pertaining to data quality can be made only through experience and training. MOODS makes the data available to anyone with a knowledge of computers, so there is a danger that the data will be used by individuals with little oceanographic training.

An effort has been made to flag questionable data upon inclusion into MOODS. Data are flagged as questionable for over land, maximum depth errors, platform speed errors, sea-surface temperature (SST) range errors, platform call-sign discrepancies, and duplicate profiles. Profiles are flagged as duplicates or near duplicates of each other because of the multipath nature of data that enter MOODS (for example, duplicates can occur when the same data reside in several MOODS source locations); however, erroneous data and duplicate profiles do slip in unflagged. By using only unflagged data, the user is only partially protected from erroneous data. Quality control is becoming critical as more users begin to rely on MOODS. As this file, which is accessible to the oceanographic community, becomes more and more used, a critical need for additional quality control of the data set has arisen. This report describes an editor that can be used for editing data contained in MOODS. The editor is not specifically tailored to MOODS, and thus can also be used for editing other data bases with similar structure. All the computer programs that make up the editor are written in FORTRAN 77. The computer program package for the editor is described in Appendix A and listed in Appendix C. In Appendix B, usage of the editor is discussed in conjunction with implementation of MOODS on a microcomputer.

Editing approach

The editor was designed for use during extracting or updating the MOODS data base. The editor is designed to be flexible to changing conditions; that is, a set of default values are suitable for many of the editor utilizations, but the user maintains flexibility to tailorize the defaults for specific data. Data output is in the same format as input. An error log is produced so that profile rejections are tabulated and later checked, if desired.

The editor behaves as if it were a series of filters. Profiles that enter the filter series are discarded if they do not pass various editor tests. For error conditions that are not fatal, an attempt is made to repair the profile, and then the profile must pass the other editor checks. The following is a list of editor checks, which can be turned

on or off (with the exceptions of 4, 5, and 6), and default edit parameters, which can be changed by the user:

1. MOODS error flag—duplicate profiles discarded.
2. Over land—profile discarded.
3. Depth bounds—profile discarded if maximum depth exceeds 6500 m (default).
4. Unsorted depths—depths are sorted and data retained.
5. Duplicate depths—data at duplicate depths removed and only last set kept.
6. Negative depths—depths changed to positive and retained.
7. Depths below the bottom—profile discarded if depth exceeds local bathymetry by 1% (default).
8. Temperature bounds—profile discarded if data falls outside the range from −2.5°C to 32.0°C (defaults) from the surface to 1000 m (default). Then the temperature limits change from −2.5°C to 12.0°C (defaults).
9. Salinity bounds—profile discarded if data falls outside the range from 33.0 to 37.0 ppt (defaults) from the surface to 1000 m (default). Then the salinity limits change from 34.0 to 36.0 ppt (defaults).
10. Temperatures all zero—profile discarded.
11. Salinities all zero—salinities discarded.
12. Temperature inversion—profile discarded for inversion greater than 1°C (default) between the surface (default) and the bottom of the profile. The user can set the first default for more stringent editing, and the second default for ignoring near-surface inversions that can occur.
13. Density inversion—profile discarded for in situ density inversion greater than $10^{**}-5$ gr/cm**3 (default) between the surface (default) and the bottom of the profile.
14. Misplaced profiles—profile discarded if sea surface temperature differs by more than 3.5 standard deviations (default) from mean sea surface temperature of the subset of MOODS that is being edited. Assume that either the location or month is wrong.
15. Duplicate profiles—shallower profiles discarded if two or more profiles are within ±5 km and ±30 minutes of each other. This test finds the same data that have entered the base by different sources; it also finds closely spaced time-series data.

Test applications

Four data extractions were made from the MOODS data base to test the editor. The data sites were Northwest Atlantic—January (2831 profiles) and Northeast Atlantic—January (1076 profiles) (Fig. 1), and Northeast Pacific—July (825 profiles) and Northeast Pacific—January (889 profiles) (Fig. 2). Results are presented in Table 1. The total percentage of rejections ranged from 17% in the Northeast Pacific in July to 39% in the Northwest Atlantic in January. The larger number of rejections in the Northwest Atlantic can perhaps be attributed to a greater amount of ship-of-opportunity observations (with less quality control) in this region than in the other test regions.

Temperature and density inversions (Fig. 3, for example) accounted for the majority of profile rejections by the ocean limits test. These inversions result from one or more erroneous temperatures or salinities in a profile. Edit parameters were more rigorously chosen here than when the data were edited upon initial inclusion in MOODS.

Misplaced profiles, profiles with incorrect geographical positions or incorrect dates, accounted for about 1% of the rejections. Most were probably a result of transposed digits in the profile positions or dates contained in the header. Each rejection in all four test sets was verified to confirm that the test excluded obvious misplaced profiles.

A significant percentage of duplicate profiles not already flagged in MOODS (4–16%) was also found. In most cases these duplicates were copies of profiles that were digitized at different levels, cut off at a shallower depth, or contained only sea-surface temperatures. The percentages of profiles flagged as duplicates as a function of time and space are summarized for the Northwestern Atlantic test case in Table 2. For the default values of ±5 km and ±30 minutes, 16% of the profiles were flagged as duplicates. These defaults were scrutinized by manually checking every flagged duplicate in all four test sets.

Changing these duplicate check defaults could be useful in several other applications. For example, the data could be thinned when independent rather than clumped observations are required for a statistical application. Also, this feature of the editor could be used in the opposite manner to seek out (via the log file) all closely spaced time-series data in a region.

Conclusions

These additional quality control measures will be valuable during the updating of the MOODS data base. The existing data base contains a significant amount (17–39% for the four test sets) of bad data. Duplicate profiles add overhead to the management of the data base and bias data statistics. For MOODS updates or extractions, we recommend that the data be passed through an editor such as this one in which the edit parameters can be tailored. Passing the entire MOODS data base through

the editor and consequently rebuilding the data base should also be considered.

The editor described here can be used on most computers, and since it is not tailored specifically for the MOODS data base, can be used on other data bases. The data format output by the editor is identical to the data input format, and the program is flexible and user friendly. Default editor parameters can be changed readily by the user. Profile rejections are tabulated in an error log file, which also can serve as a guide for tightening or loosening editor default parameters.

Bibliography

Bauer, R. A. (1985). *Functional Description Master Oceanographic Data Set (MOODS), Documentation Report*. Compass Systems, Inc., San Diego, California.

Bauer, R. A. (1983). *Users Guide for Univac 1180 Master Oceanographic Observation Data Set Access Routines*. Compass Systems, Inc., San Diego, California.

Kassoff, L. E. (1983). *MOODS Users's Guide, STD-N-214*. John Hopkins University Applied Physics Laboratory, Laurel, Maryland.

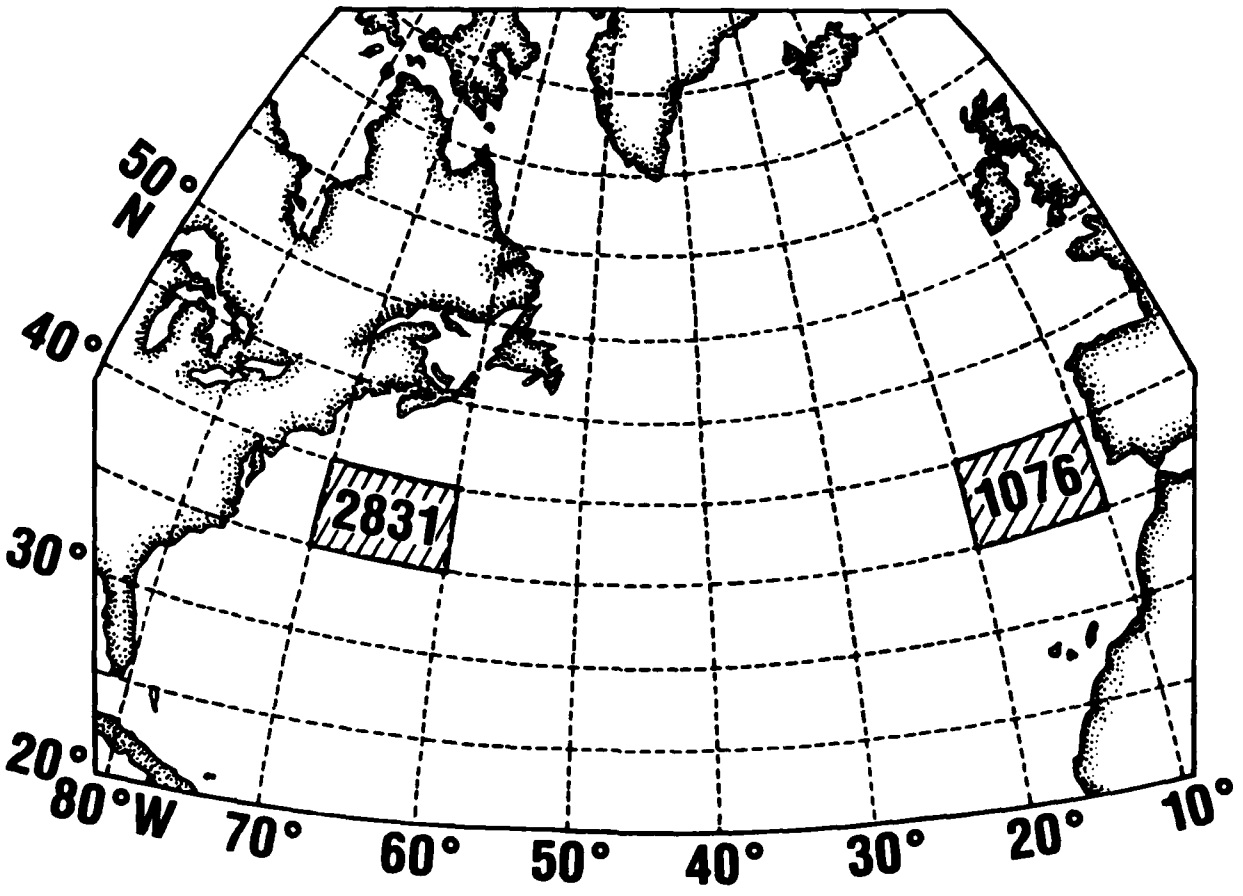


Figure 1. The editor was tested with 1076 January profiles obtained from a region off the west coast of Africa and 2821 January profiles obtained from a region off the east coast of the United States

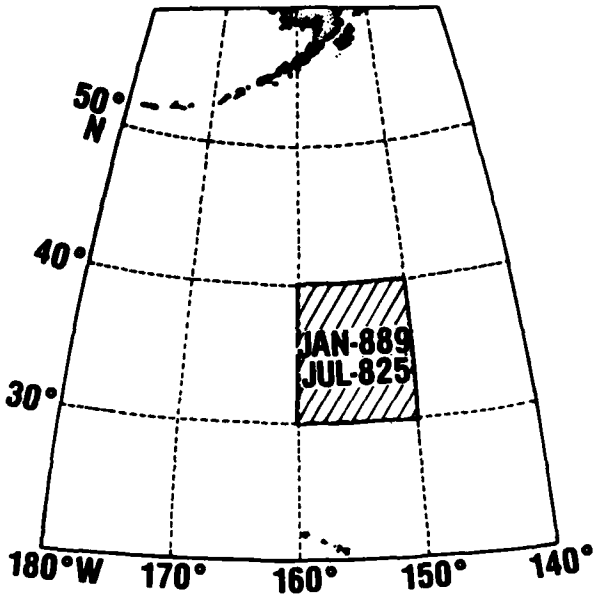


Figure 2. The editor was tested with 889 January profiles and 825 July profiles obtained from a region in the northeast Pacific Ocean.

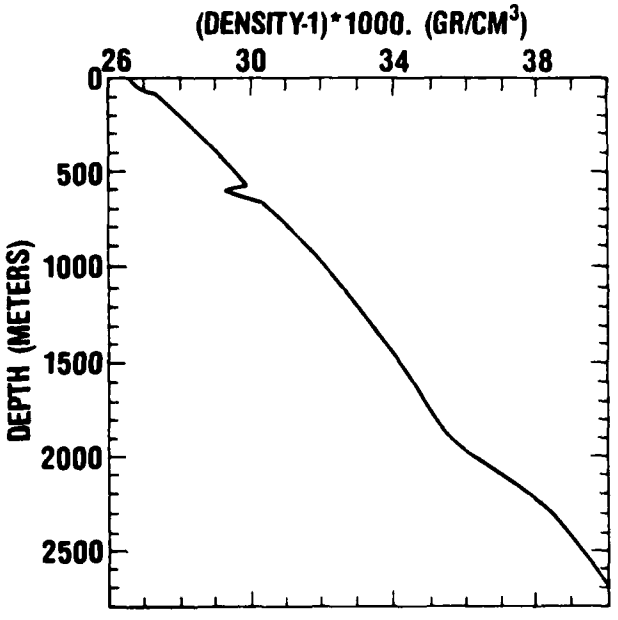
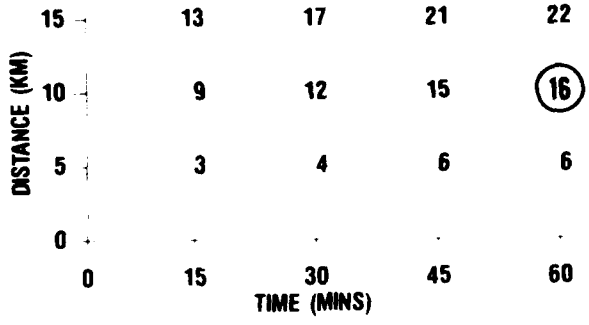


Figure 3. The editor will reject this profile due to the density inversion occurring near 600 m.

Table 1. Summary of profile rejections.

Data Set	OBS	Duplicate Profiles Flagged by Woods Error Code	Failed Oceanographic Limit Tests	Additional Duplicate Profiles	Misplaced Profiles	% Total
W Coast Africa (JAN)	1076	16	2	4	1	23
NE Pacific (JUL)	825	9	2	6	0	17
NE Pacific (JAN)	889	13	5	7	0	25
East Coast USA (JAN)	2821	9	13	16	1	39

Table 2. Percentages of profiles flagged as duplicates, as a function of time and distance between profiles, are summarized for the Northwestern Atlantic test case.



Appendix A: Editor input control

APPENDIX A

EDITOR INPUT CONTROL

All programs in the editor package were written in standard ASCII FORTRAN and were tested on VAX 11/750 and UNIVAC 1180 computers. The editor takes output from the MOODGET routine (located in the program file MOODS*PROGRAMS, on the NAVOCEANO UNIVAC, and DRB0:[BILL.MOODS] on the NORDA Code 331 VAX) as input and eliminates the bad or questionable profiles. Output from the editor, which consists of profiles written in the same format as the input profiles, is readable with the text editor. A summary detailing which profiles have been removed and why is given in a log file, "INPUTFILENAME*LOG" (UNIVAC) or "INPUTFILENAME.LOG" (VAX).

The main program, MOODSED, calls DEPTHC, DUPGET, GMLIC, INVERC, NODAYS, RDIN, RDMOOD, SOLIC, SORT, STATS, VALUEC, WEMOOD, DB2DEP(UNIVAC), and LNDCHK (UNIVAC) or LANDMASK (VAX). The subroutine (DB2DEP), which checks the depth of the bottom, and the corresponding bathymetric data file were not available on the VAX.

Program input is entered by instructions or prompted via questions by typing "HELP" upon program execution. Instructions, which are summarized below, may be entered in any order.

Editor instructions:

IFIL	input filename (default MOODINS).
OFIL	output filename (default MOODSOUT).
TMIN	minimum temperature allowed (default -2.5).
TMAX	maximum temperature allowed (default 32.0).
SMIN	minimum salinity allowed (default 32.0).
SMAX	maximum salinity allowed (default 40.0).
TL01	minimum temperature allowed past ZMIN (default -2.5).
TH11	maximum temperature allowed past ZMIN (default 12.0).
SL01	minimum salinity allowed past ZMIN (default 34.0).
SH11	maximum salinity allowed past ZMIN (default 36.0).
ZMIN	depth at which to change limits, TL01, etc. (default 1000).
ZMAX	maximum depth allowed (default 6500).
LCHK	set to 1 to skip land checking (default 0).
IDCK	set to 1 to skip neg. and non-mon. depth checks (default 0).
IECK	set to 1 to skip MOODS dup profile error code checking (default 0).
IBCK	set to 1 to skip depth thru bottom checking (default 0).
ITOC	set to 1 to skip checking for all-zero temps (default 0).
ISOC	set to 1 to skip checking for all-zero salinities (default 0).
ISBC	set to 1 to skip checking for bad salinities (default 0).
ITBC	set to 1 to skip checking for bad temperatures (default 0).
IVCK	set to 1 to skip checking for density/temp inversions (default 0).
ZIVC	check for inversions beyond this depth (default 0.0).
DTOL	tolerance for density inversion check (default 0.00001).
TTOL	tolerance for temperature inversion check (default 1.0).
BTOL	through bottom depth check tolerance, % (default 1.0 %).
IMIS	set to 1 to skip checking for misplaced profiles (default 0).
DMIS	maximum depth of near SST for misplaced profile check (default 20.0).
STOL	standard deviation tolerance for misplaced prof check (default 3.5).
IDUP	set to 1 to skip duplicate profile check (time-dist. window) (default 0).

```
IMIN    time window in minutes for dup. prof check (default 30).
DIST    distance window in km for dup. prof check (default 5.0).
C       comment statement, no action.
@END    ends instructions and begins program execution (UNIVAC).
CTRL Z  "CTRL Z" ends instructions and begins program execution (VAX).
```

Example run (UNIVAC) with instruction file input:

```
@RUN MOODSED
C    DEFINE INPUT FILENAME
IFIL MOODSDAT
C    DEFINE OUTPUT FILENAME
OFIL OUTFILE
C    IGNORE ALL ZERO SALINITIES
ISOC 1
C    SKIP LAND CHECKING
LCHK 1
@END
```

Questions are solicited upon typing "HELP". A blank return maintains the default value. Program execution begins upon answering last question or by typing "@END for UNIVAC execution or "CTRL Z" for VAX execution. The questions are summarized below.

```
INPUT FILENAME?
OUTPUT FILENAME?
CHECK FOR BAD TEMPERATURES?  DEFAULT IS YES
MINIMUM TEMPERATURE (DEG C) ALLOWED IS   -2.500000
NEW MINIMUM TEMPERATURE?
MAXIMUM TEMPERATURE (DEG C) ALLOWED IS    30.00000
NEW MAXIMUM TEMPERATURE?
CHECK FOR BAD SALINITIES? DEFAULT IS YES
MINIMUM SALINITY ALLOWED IS    33.00000
NEW MINIMUM SALINITY?
MAXIMUM SALINITY ALLOWED IS    38.00000
NEW MAXIMUM SALINITY?
MAXIMUM DEPTH (M) ALLOWED IS    6500.000
NEW MAXIMUM DEPTH?
CHECK FOR TEMP/DENSITY INVERSIONS?  DEFAULT IS YES
MAXIMUM TEMP. INVERSION (DEG C) ALLOWED IS    1.000000
NEW MAXIMUM TEMPERATURE INVERSION?
MAXIMUM DENSITY INVERSION (GM/CM**3) ALLOWED IS   9.9999997E-06
NEW MAXIMUM DENSITY INVERSION?
CHECK FOR INVERSION BEYOND DEPTH   0.0000000E+00 M
NEW DEPTH?
CHECK FOR DEPTHS GREATER THAN BOTTOM?  DEFAULT IS YES
BOTTOM CHECKING - PERMISSIBLE DEPTH ERROR (%)    1.000000
NEW DEPTH ERROR (%)?
CHECK FOR DATA OVER LAND?  DEFAULT IS YES
CHECK FOR NEGATIVE, AND NON-MONOTONIC DEPTHS?
DEFAULT IS YES
CHECK MOODS DUPLICATE PROFILE ERR CODE?  DEFAULT IS YES
CHECK IF TEMPERATURES ARE ALL ZERO?  DEFAULT IS YES
CHECK IF SALINITIES ARE ALL ZERO? DEFAULT IS YES
```

```
CHECK FOR MISPLACED PROFILES? DEFAULT IS YES
MAX STANDARD DEVIATION IS    3.500000
NEW MAX STANDARD DEVIATION?
MAX DEPTH OF NEAR SURFACE TEMP IS    20.00000     M
NEW MAX DEPTH?
CHECK FOR DUPLICATE PROFILES? DEFAULT IS YES
MAX TIME IN MINUTES BETWEEN PROFILES IS          60
NEW MAX TIME?
MAX DISTANCE IN KM BETWEEN PROFILES IS    10.00000
NEW MAX DISTANCE?
CHANGE TEMP AND/OR SAL LIMITS AT DEPTH? DEFAULT IS NO
NEW DEPTH FOR CHANGING TEMP/SAL LIMITS? DEFAULT IS    6500.000
ENTER NEW TEMPERATURE/SALINITY LIMITS
MINIMUM TEMPERATURE (DEG C) ALLOWED IS   -2.500000
NEW MINIMUM TEMPERATURE?
MAXIMUM TEMPERATURE (DEG C) ALLOWED IS    12.00000
NEW MAXIMUM TEMPERATURE?
MINIMUM SALINITY ALLOWED IS    34.00000
NEW MINIMUM SALINITY?
MAXIMUM SALINITY ALLOWED IS    37.00000
NEW MAXIMUM SALINITY?
```

Appendix B: MOODS on a micro or personal computer

APPENDIX B

MOODS ON A MICRO- OR PERSONAL COMPUTER

A micro- or personal computer presents two difficulties for MOODS: data storage is limited and much of the existing MOODS software is not adaptable to microcomputer usage. However, these problems are not insurmountable.

The following system requirements will implement MOODS on a microcomputer.

- o The system must be user friendly.
- o The system should require a minimum of maintenance.
- o The system must be flexible, easy to use in new ways, and easily adaptable to new requirements.
- o The system should be as machine independent as possible. All software should be written in machine-independent languages, such as FORTRAN 77, wherever possible and should be reasonably portable to other systems. Portability problems are most likely to occur in the input/output (I/O) area. Structured programs with I/O isolated to one or a few subroutines are easier to convert to other systems than programs with I/O statements embedded throughout.

MOODS presently contains approximately 3.5 million profiles and is growing yearly. Estimating about 100 data words (4 bytes/word) per profile, MOODS now contains about 1.4 gigabytes of data. If 25% of the data is removed through the editor, about one gigabyte of data still remains. For a double-sided, double-density removable floppy disk with a storage capacity of about 1 megabyte, about 1000 floppy disks would be required to hold the present MOODS. For a Winchester drive with 100 megabytes of hard disk storage, 10 hard disks would be required to hold the present MOODS. Optical disks, which are just now coming onto the market, offer the extremely high storage density of 1 gigabyte on a single 12-inch platter. One optical disk could hold the present MOODS. Magnetic tape, which tends to be used for backup rather than primary storage, is not practical because of slow random access.

Maintaining the entire MOODS on floppy disks is impractical. A possible mechanism for reducing the size of MOODS is the subsetting of the data set. Subsets of MOODS can be defined by region, season, source, and depth. These subsets of MOODS could then be loaded onto floppy disks for particular applications. Even larger subsets of MOODS could be loaded onto hard disks, but then the ruggedness, cheapness, and simplicity of floppy disk systems are lost (a dedicated Winchester drive and a magnetic tape drive for backup of the Winchester disk are required). Optical disk technology is not yet sufficiently developed and proven. Furthermore, MOODS is growing rapidly; thus, the storage requirements increase.

Another mechanism for reducing the size of MOODS to a manageable quantity for microcomputer applications involves thinning the data set, that is, to reduce the number of profiles in highly sampled regions to several representative profiles. Floppy disks are then practical. Biasing the data by the highly sampled areas would also be reduced through thinning. Data sets could be thinned for the particular applications. The data base editor described in this report can be

used for thinning profiles by setting the parameters for the duplicate profile check accordingly.

Software for manipulating MOODS is available at the Fleet Numerical Oceanography Center for a CDC computer (Bauer, 1981), at the Naval Oceanographic Office for a UNIVAC computer (Bauer, 1983), and at the John Hopkins University Applied Physics Laboratory for a VAX computer (Kassoff, 1983). The CDC programs tend to be large and complex due to their batch processing environment. The UNIVAC programs are similar, since they tend to be converted CDC or similarly designed programs. Modifying these programs should be quite difficult and often impossible because of their complexity and machine dependence. Entirely different programs were written for the VAX but similar problems can be anticipated.

Existing micro/personal computer data-base systems need to be identified for which MOODS applications can be designed to properly fit and to take advantage of their data-base capabilities and facilities. Either actual or interpolated data values for all products (where applicable) must be provided by the system. Useful products that should be available from the microcomputer MOODS follow.

Profile plots
TS plots
Data listings
Minimums, maximums, means, and standard deviations for data values
Data distributions
Waterfall plots
Composite profile plots
Computed sound velocity profiles
Mean profiles
Vertical cross sections
Horizontal cross sections
Dynamic heights/geostrophic currents
Inversion/strength mapping
Mixed layer depth/strength mapping
Acoustic duct depth/strength mapping

Appendix C: Program listings

APPENDIX C

PROGRAM LISTINGS

```
C*********************************************************************
C       PROGRAM:  MOODSED
C       PURPOSE:  TO EDIT OUTPUT OF MOODS DATA BASE
C
C         This program takes output from the MOODGET routine and removes
C         the bad or questionable profiles; this output is in the same
C         format as the output from the MOODGET routine.  The output is
C         readable with the text editor.  A summary of which profiles
C         are removed is given in "INPUTFILENAME.LOG".
C
C         NOTE:  If another data access routine is substituted
C         for MOODGET, then subroutines RDMOOD and WEMOOD must be
C         changed accordingly.
C
C         UNIVAC:  SUBROUTINES CALLED:      LNDCHK
C                                           RDIN
C                                           DEPTHC
C                                           VALUEC
C                                           INVERC
C                                           RDMOOD
C                                           WEMOOD
C                                           SOLIC
C                                           STATS
C                                           NODAYS
C                                           DUPGET
C                                           GMLIC
C                                           SORT
C                                           DB2DEP
C
C         VAX:  LANDMASK is substituted for LNDCHK.
C               There is no bottom checking routine (DB2DEP).
C
C         UNIVAC map element: MAPMOODSED
C         VAX link element: MOODSED.LNK
C
C         UNIVAC sample run: RUNMOODSED
C         VAX sample run: MOODSED.RUN
C
C         Sample input MOODS data, UNIVAC: .MOODAT
C          Convert program element to SDF file.
C          @ASG,UP DATA.
C          @DATA,I DATA.
C          @ADD,D DATA.
C          @END
C
C         Sample input data, VAX: MOODAT.DAT
C
C       INPUT: INSTRUCTION FILE FORMAT
C         IFIL  input filename (default [BILL.MOODS]MOODS)
```

```
C         OFIL  output filename (def MOODS)
C         TMIN  minimum temperature allowed (def -2.5)
C         TMAX  maximum temperature allowed (def 32.0)
C         SMIN  minimum salinity allowed (def 32.0)
C         SMAX  maximum salinity allowed (def 40.0)
C         TLO1  minimum temperature allowed past ZMIN (def -2.5)
C         THI1  maximum temperature allowed past ZMIN (def 12.0)
C         SLO1  minimum salinity allowed past ZMIN (def 34.0)
C         SHI1  maximum salinity allowed past ZMIN (def 36.0)
C         ZMIN  depth at which to change limits, TLO1, etc. (def 1000.)
C         ZMAX  maximum depth allowed (def 6500.0)
C         LCHK  set to 1 to skip land checking (def 0)
C         IDCK  set to 1 to skip neg. and non-mon. depth checks (def 0)
C         IECK  set to 1 to skip MOODS dup profile error code checking (def 0)
C         IBCK  set to 1 to skip depth thru bottom checking (def 0)
C         ITOC  set to 1 to skip checking for all-zero temps (def 0)
C         ISOC  set to 1 to skip checking for all-zero salinities (def 0)
C         ISBC  set to 1 to skip checking for bad salinities (def 0)
C         ITBC  set to 1 to skip checking for bad temperatures (def 0)
C         IVCK  set to 1 to skip checking for density/temp inversions
C               (def 0)
C         ZIVC  check for inversions beyond this depth (def 0.0)
C         DTOL  tolerance for density inversion check (def 0.00001)
C         TTOL  tolerance for temperature inversion check (def 1.0)
C         BTOL  through bottom depth check tolerance, % (def 1.0 %)
C         IMIS  set to 1 to skip checking for misplaced profiles (def 0)
C         DMIS  max depth of near SST for misplaced profile check (def 20.0)
C         STOL  standard dev. tolerance for misplaced prof check (def 3.5)
C         IDUP  set to 1 to skip dup. prof. check (time-dist. window) (def 0)
C         IMIN  time window in minutes for dup. prof check (def 30)
C         DIST  distance window in km for dup. prof check (def 5.0)
C         C     comment statement, no action.
C         @END  ends instructions and begins program execution (UNIVAC)
C         Z     "CTRL Z" ends instructions and begins program execution (VAX)
C
C         HELP  program solicits input
C
C         EXAMPLE RUN WITH INSTRUCTION FILE INPUT:
C                   $ RUN [BILL.MOODS]MOODSED
C                   C    DEFINE INPUT FILENAME
C                   IFIL MOODS.DAT
C                   C    DEFINE OUTPUT FILENAME
C                   OFIL OUTFILE.DAT
C                   C    IGNORE ALL ZERO SALINITIES
C                   ISOC 1
C                   C    SKIP LAND CHECKING
C                   LCHK 1
C                   Z
C
C*************************************************************************
C
C   THIS PROGRAM CAN HANDLE A MAXIMUM OF IPMAX PROFILES, WITH A MAXIMUM
C   OF LEVMAX LEVELS FOR EACH PROFILE.
C
```

```
      PARAMETER (IPMAX=99000,LEVMAX=1000)
C
      CHARACTER*13 GL1,GL2
      CHARACTER*72 OUTFILE,INFILE,LOGFILE
      CHARACTER*131 INPUT,INPUTS
      CHARACTER*10 IP
C
      COMMON /VALIM /TMIN,TMAX,SMIN,SMAX,TEMP,SAL,
     &              TLO1,THI1,SLO1,SHI1,ZMIN
      COMMON /FILES/ INFILE,OUTFILE
      COMMON /CHKS/ LCHK,IDCK,ITOC,ISOC,ISBC,ITBC,IVCK,ZIVC,IECK,IBCK,
     &              IMIS,DMIS,STOL,IDUP,IMIN,DIST
      COMMON /ZLIM/ ZMAX,ZVAL
      COMMON /INVERT/ DTOL,TTOL,BTOL,DELDEN,DELT
      COMMON /RDCOM/ F(10),ID(10),TOP,BOT,IS
      COMMON /RDCOMA/ INPUTS,GL1,GL2,IP
C
      DIMENSION D(LEVMAX),T(LEVMAX),S(LEVMAX)
      DIMENSION IPCT(12),ISCODE(30),INVFLG(2),ISCODG(30)
      DIMENSION TIME(IPMAX),INO(IPMAX),IDUPER(IPMAX)
C
      EQUIVALENCE (IDUPER,TIME)
C
      DATA ITOTCT,IPASS,IDCT/0,1,1/
C
C*********************************************************************
C   READ INPUT
C
C   SAVE MAX NO. OF LEVELS FOR CHECK IN RDMOOD
      MAXLEV=LEVMAX
C
      WRITE(6,*)' ENTER INSTRUCTIONS OR --'
      WRITE(6,*)' TYPE "HELP" FOR INPUT SOLICITATION'
C
C   READ INPUT INSTRUCTIONS FROM INSTRUCTION FILE
      CALL RDIN(ISOL)
C
C
C   SOLICITATE INPUT
      IF(ISOL.EQ.1)CALL SOLIC
C
C*********************************************************************
C   OPEN FILES
      OPEN(UNIT=24,ACCESS='DIRECT',FORM='UNFORMATTED',
     &   STATUS='SCRATCH',ERR=9089,RECL=3)
      OPEN(UNIT=26,STATUS='SCRATCH',ERR=9091)
      OPEN(UNIT=27,FILE=INFILE,STATUS='OLD',READONLY,ERR=9092)
      OPEN(UNIT=21,FILE=OUTFILE,STATUS='NEW',ERR=9093)
C
      K=72
      DO 10 I=1,72
      IF(INFILE(K:K).NE.' ')GO TO 11
10    K=K-1
11    CONTINUE
```

```
C
         LOGFILE=INFILE(1:K)//'.LOG'
         OPEN(UNIT=29,FILE=LOGFILE,STATUS='NEW',ERR=9094)
         WRITE(29,*)'                MOODSED ERROR LOG'
         WRITE(29,*)
         GO TO 3
C
C*********************************************************************
C
C    IF DUPLICATE OR MISPLACED PROFILE CHECK IS ON, THEN A SECOND PASS
C    THROUGH THE DATA IS REQUIRED
C
C222 SECOND PASS CODE 22222222222222222222222222222222222222222222222
C    IF MISPLACED OR DUP. PROFILE CHECK IS ON THEN KEEP TRACK OF
C    BAD PROFILES
 1       IF(IPASS.EQ.1)THEN
          IF(IMIS.EQ.0.OR.IDUP.EQ.0)THEN
           WRITE(26,*)IS
           GO TO 3
          END IF
         END IF
C22222222222222222222222222222222222222222222222222222222222222222222
C
C*********************************************************************
C    KEEP TALLY OF SOURCE CODES FOR BAD DATA
         IF(ID(4).LT.30)ISCODE(ID(4))=ISCODE(ID(4))+1
         IF(ID(4).GT.29)WRITE(6,*)'SOURCE CODE GREATER THAN 29 - ',
     &   ID(4),' ',GL1,GL2,' PROF',IS
         GO TO 3
C    KEEP TALLY OF SOURCE CODES FOR GOOD DATA
 2       IF(ID(4).LT.30)ISCODG(ID(4))=ISCODG(ID(4))+1
         IF(ID(4).GT.29)WRITE(6,*)'SOURCE CODE GREATER THAN 29 - ',
     &   ID(4),' ',GL1,GL2,' PROF',IS
C    READ MOODS DATA
 3      CALL RDMOOD(D,T,S,NLEV,MAXLEV,IEOF)
C
        IF(IEOF.EQ.1)GO TO 900
        IF(IEOF.EQ.2)GO TO 9000
C*********************************************************************
C
C222 MORE SECOND PASS CODE 22222222222222222222222222222222222222222222
C    SKIP MISPLACED AND DUP. PROFILE CHECKS ON PASS 1
        IF(IPASS.EQ.1)GO TO 20
C    THIS IS THE SECOND PASS THROUGH THE DATA.
C
C    CHECK TO SEE IF MORE BAD PROFILES
        IF(IBEOF.EQ.1)GO TO 15
C    SKIP BAD PROFILES
         IF(IS.EQ.IBADNO)THEN
          IF(IBEOF.EQ.0)READ(26,*,END=14,ERR=9090)IBADNO
          GO TO 1
 14       IBEOF=1
          GO TO 1
         END IF
```

```
CCC
C   APPLY CHECK FOR MISPLACED PROFILES.
 15     IF(IMIS.EQ.0)THEN
CCC
         IF(D(1).GT.DMIS)GO TO 16
         IF(T(1).LT.-998.)GO TO 16
C
C   THROW OUT MISPLACED PROFILES
         IF(T(1).GT.SSTMAX.OR.T(1).LT.SSTMIN)THEN
          WRITE(29,*)'MISPL. PROF., SST= ',T(1),' ',GL1,GL2,
     &     ' PROF ',IS
          IPCT(11)=IPCT(11)+1
          GO TO 1
         END IF
CCC
       END IF
CCC
C*********************************************************************
C   THROW OUT DUPLICATE PROFILES
C   IF NO MORE DUP. PROFILES SKIP FOLLOWING
 16    IF(IDUPCT.EQ.0)GO TO 19
       IF(IDCT.GT.IDUPCT)GO TO 19
C
       IF(IDUP.EQ.0)THEN
C   PROFILE MAY HAVE BEEN REMOVED ALREADY BY MISPLACED PROF TEST
        IF(IDUPER(IDCT).LT.IS)THEN
         IDCT=IDCT+1
         IF(IDCT.GT.IDUPCT)GO TO 19
        END IF
C
        IF(IDUPER(IDCT).EQ.IS)THEN
         IDCT=IDCT+1
         WRITE(29,*)'DUPLICATE PROFILE ',GL1,GL2,' PROF ',IS
         IPCT(12)=IPCT(12)+1
         GO TO 1
        END IF
CC
       END IF
CC
C
C   WRITE DATA ON SECOND PASS
 19    CALL WEMOOD(D,T,S,NLEV,IEOF)
       IF(IEOF.EQ.2)GO TO 9004
       GO TO 2
C22222222222222222222222222222222222222222222222222222222222222222
C
C*********************************************************************
C   COUNT TOTAL NO. OF PROFILES
 20    ITOTCT=ITOTCT+1
C   DO NOT ALLOW MORE THAN IPMAX PROFILES
       IF(ITOTCT.GT.IPMAX)THEN
        WRITE(6,*)'MAX NO. OF PROFILES PROCESSED - ',IPMAX
        WRITE(29,*)'MAX NO. OF PROFILES PROCESSED - ',IPMAX
        GO TO 900
```

```
      END IF
C*********************************************************************
C
C*********************************************************************
C   CHECK MOODS ERR CODE FOR DUPLICATE PROFILES.
C    IF NON-ZERO THEN SKIP PROFILE
       IF(IECK.EQ.1)GO TO 30
       IF(ID(5).NE.0.OR.ID(6).NE.0)THEN
        WRITE(29,*)'NON-ZERO MOODS ERR CODE ',GL1,GL2,' PROF',IS
        IPCT(1)=IPCT(1)+1
        GO TO 1
       END IF
   30  CONTINUE
C*********************************************************************
C
C*********************************************************************
C   CHECK FOR DATA OVER LAND, IF SO, THEN SKIP PROFILE
C   IX5=0 FOR WATER,IX5=1 FOR LAND
       IF(LCHK.EQ.1)GO TO 40
C
C   LAND CHECKING IN NOT AVAILABLE BEYOND 72N AND 72S.
       IF(ABS(F(1)).GT.72.0)THEN
        WRITE(6,*)' LAND CHECKING IS NOT AVAILABLE BEYOND 72N AND 72S'
        GO TO 40
       END IF
C
       IX1=F(1)
       IX2=(F(1)-IX1)*60.
       IX3=F(2)
       IX4=(F(2)-IX3)*60.
       CALL LANDMASK(IX1,IX2,IX3,IX4,IX5)
       IF(IX5.EQ.1)THEN
        IPCT(2)=IPCT(2)+1
       WRITE(29,*)'DATA OVER LAND ',GL1,GL2,' PROF ',IS
       GO TO 1
       END IF
  40   CONTINUE
C
C*********************************************************************
C
C*********************************************************************
C
C   CHECK DEPTHS FOR NEGATIVE DEPTHS, NON-MONOTONIC DEPTHS,
C   AND DEPTHS EXCEEDING MAX DEPTH.
C   FIX NEGATIVE AND NON-MONOTONIC DEPTHS, THROW OUT PROFILES
C   IF MAX DEPTH IS EXCEEDED.
C
       IF(IDCK.EQ.1)GO TO 50
       CALL DEPTHC(D,T,S,NLEV,IZFLG,IDFLG)
C
C   IF IDFLG=0 THEN DEPTHS ARE OK, 1 FOR NEG. DEPTHS, 2 FOR NON-MON.
C   IZFLG=1 THEN MAX DEPTH EXCEEDED THROW THESE PROFILES OUT
       IF(IDFLG.GT.0)THEN
        IF(IDFLG.EQ.1)WRITE(29,*)'NEG. DEPTHS ENCOUNTERED, ASSUMED POS.'
```

```
     *  ,GL1,GL2,' PROF ',IS
        IF(IDFLG.EQ.2)WRITE(29,*)'NON-MON. DEPTHS FOUND, ',
     *  'MADE MON. ',GL1,GL2,' PROF ',IS
        IF(IDFLG.EQ.3)WRITE(29,*)'NEG. AND NON-MON. DEPTHS FOUND '
     *  ,' FIXED ',GL1,GL2,' PROF ',IS
       END IF
       IF(IZFLG.EQ.1)THEN
        WRITE(29,*)'MAX DEPTH EXCEEDED, PROFILE REMOVED, Z= ',
     *  ZVAL,GL1,GL2,' PROF ',IS
        IPCT(3)=IPCT(3)+1
        GO TO 1
       END IF
  50   CONTINUE
C
C*****************************************************************************
C
C*****************************************************************************
C   CHECK FOR DEPTHS EXCEEDING BOTTOM DEPTH (TOLERANCE OF BTOL %)
C   IF SO, THROW PROFILE OUT.
C   SYNBAPS DATA BASE IS USED.
       IF(IBCK.EQ.1)GO TO 55
CW
       WRITE(6,*) ' BOTTOM CHECKING NOT AVAILABLE ON VAX'
       IBCK=1
       IF(IBCK.EQ.1)GO TO 55
CW
        X1=F(1)
        X3=F(2)
C       CALL DEPTH(X1,X3,BDEPTH)
       BDP=BDEPTH+(BDEPTH*BTOL*.01)
       DO 52 I=1,NLEV
       IF(D(I).GT.BDP)THEN
        WRITE(29,*)'BOTTOM/DEPTH',BDEPTH,'/',D(I),GL1,GL2,' PROF',IS
        IPCT(4)=IPCT(4)+1
        GO TO 1
       END IF
  52   CONTINUE
C*****************************************************************************
C
C*****************************************************************************
C   CHECK FOR IMPOSSIBLE T VALUES AND IMPOSSIBLE S VALUES
C   IF PROFILE HAS A SINGLE BAD VALUE, THROW PROFILE OUT
 55     CALL VALUEC(D,T,S,NLEV,IVFLG)
C   IF IVFLG=0 THEN DATA ARE OK, 1 FOR SAL ARE ALL ZERO,
C   2 FOR TEMP ARE ALL ZERO, 3 FOR TEMP AND SAL ARE ALL ZERO,
C   4 FOR BAD TEMP VALUE, 5 FOR BAD SAL VALUE, 6 FOR BAD TEMP AND SAL VAL,
C   7 FOR BAD SAL VALUE AND TEMP ARE ALL ZERO,
C   8 FOR BAD TEMP VALUE AND SAL ARE ALL ZERO
C
C   THROW OUT PROFILE IF BAD TEMP AND ITBC=0
C   KEEP BAD TEMP IF ITBC=1
        IF(ITBC.EQ.1)GO TO 60
        IF(IVFLG.EQ.4.OR.IVFLG.EQ.6.OR.IVFLG.EQ.8)THEN
        IPCT(5)=IPCT(5)+1
```

```
        WRITE(29,*)'BAD TEMP VALUE OF ',TEMP,' ',GL1,GL2
     *    ,' PROF ',IS
          GO TO 1
          END IF
 60       CONTINUE
C    THROW OUT PROFILE IF BAD SAL AND ISBC=0
C    KEEP BAD SAL IF ISBC=1
          IF(ISBC.EQ.1)GO TO 62
          IF(IVFLG.EQ.5.OR.IVFLG.EQ.6.OR.IVFLG.EQ.7)THEN
          IPCT(6)=IPCT(6)+1
          WRITE(29,*)'BAD SAL VALUE OF ',SAL,' ',GL1,GL2
     *    ,' PROF ',IS
          GO TO 1
          END IF
 62       CONTINUE
C    CHANGE SALINITIES TO MISSING (-999) WHEN SALINITIES
C    ARE ALL 0 IF ISOC=0.   KEEP IF ISOC=1
         IF(ISOC.EQ.1)GO TO 64
         IF(IVFLG.EQ.1.OR.IVFLG.EQ.8)THEN
          IPCT(7)=IPCT(7)+1
          WRITE(29,*)'SALINITIES ARE ALL ZERO ',GL1,GL2
     *    ,' PROF ',IS
         END IF
 64      CONTINUE
C    THROW OUT PROFILE WHEN TEMPERATURES ARE ALL 0 IF ITOC=0
C    KEEP IF ITOC=1
         IF(ITOC.EQ.1)GO TO 66
         IF(IVFLG.EQ.2.OR.IVFLG.EQ.7)THEN
          IPCT(8)=IPCT(8)+1
          WRITE(29,*)'TEMPERATURES ARE ALL ZERO ',GL1,GL2
     *    ,' PROF ',IS
          GO TO 1
         END IF
 66      CONTINUE
C
C*******************************************************************************
C
C*******************************************************************************
C    CHECK FOR DENSITY INVERSIONS (TOLERANCE DENTOL) IF SALINITY IS
C    PRESENT, OTHERWISE CHECK FOR TEMPERATURE INVERSIONS (TOLERANCE TTOL)
C    THROW PROFILES OUT IF TOLERANCE IS EXCEEDED.
C    INVFLG(1)=1 FOR DENSITY INVERSION, INVFLG(2)=1 FOR TEMP INVERSION.
         IF(IVCK.EQ.1)GO TO 70
         ZIVC1=ZIVC
         CALL INVERC(D,T,S,NLEV,ZIVC1,INVFLG)
C
C    IF INVFLG(1 OR 2)=0 THEN NO INVERSIONS EXCEEDING TOLERANCE.
C    IF DENSITY INVERSION, OR TEMPERATURE INVERSION - THEN
C    THROW THESE PROFILES OUT.
         IF(INVFLG(1).GT.0.OR.INVFLG(2).GT.0)THEN
          IF(INVFLG(1).EQ.1)IPCT(9)=IPCT(9)+1
          IF(INVFLG(1).EQ.1)WRITE(29,*)'DENSITY INVERSION-',DELDEN,' ',GL1
     *   ,GL2,' PROF',IS
          IF(INVFLG(2).EQ.1)IPCT(10)=IPCT(10)+1
```

```
        IF(INVFLG(2).EQ.1)WRITE(29,*)'TEMP INVERSION-',DELT,' ',GL1,GL2,
     *  ' PROF',IS
        GO TO 1
       END IF
 70    CONTINUE
C*************************************************************************
C
C*************************************************************************
C    CHECK FOR MISPLACED PROFILES (FIRST PASS COMPUTATIONS)
C    CALCULATE MEAN AND STANDARD DEVIATION OF THE NEAR SST FOR EXTRACTED
C    DATA SET.  NEAR SST IS THE TEMPERATURE AT DEPTHS LESS THAN DMIS.
C    IF THE SST IS GREATER THAN STOL STANDARD  DEVIATIONS FROM THE MEAN,
C    ASSUME THE PROFILE IS MISPLACED AND REMOVE THE PROFILE.
C    THIS TEST REQUIRES A PASS THROUGH THE CLEANED DATA.
        IF(IMIS.EQ.1)GO TO 74
        ISF=0
        IF(D(1).GT.DMIS)THEN
         WRITE(29,*)'PROFILE ',IS,'STARTS TO DEEP FOR MISPL. PROF. TEST'
         GO TO 74
        END IF
        DAT=T(1)
C    UPDATE STATS
        CALL STATS(ISF,DAT,VMEAN,VSDEV,NNUM)
 74     CONTINUE
C*************************************************************************
C
C*************************************************************************
C    DUPLICATE PROFILE COMPUTATIONS (PASS 1)
C    CHECK FOR DUPLICATE PROFILES BY APPLICATION OF A TIME-DISTANCE
C    WINDOW  ON THE SECOND PASS THROUGH THE DATA
C    SORT ON TIME AND THEN CHECK DISTANCE BETWEEN PROFILES WITHIN
C    THE TIME WINDOW.  SAVE TIME, LAT, LONG, PROFILE NO. AND NO. OF
C    DATA LEVELS PER PROFILE ON FIRST PASS.
       IF(IDUP.EQ.1)GO TO 78
C    COUNT NO. OF PROFILES FOR DUP TEST
      KK=KK+1
C    SAVE PROFILE NO.
       INO(KK)=IS
C    GET JULIAN DAY
       READ(GL1(1:4),5700)IYR
 5700 FORMAT(I4)
       READ(GL1(5:6),5701)IMON
 5701 FORMAT(I2)
       READ(GL1(7:8),5701)IDAY
       READ(GL1(10:11),5701)IHR
       READ(GL1(12:13),5701)IMN
       CALL NODAYS(IYR,IMON,IDAY,NDYM,NDY,IERR)
C
       IF(IERR.GT.0)THEN
        WRITE(26,*)'BAD TIME FOR PROFILE NO. ',IS,'IYR=',IYR,
     & 'IMON=',IMON,'IDAY=',IDAY
        TIME(KK)=-999.
        GO TO 77
       END IF
```

```
C
C   CHECK FOR LEAP YEARS
      IDYYR=365
      IF(MOD(IYR,4).EQ.0)IDYYR=366
  GET DECIMAL YEAR TIME
      RHRYR=IDYYR*24.
      RMINYR=RHRYR*60.
      RHR=FLOAT(IHR)
      RMIN=FLOAT(IMN)
      RYR=FLOAT(IYR)
      RDY=FLOAT(NDY)
      RDYYR=FLOAT(IDYYR)
      TIME(KK)=(RYR-1900)+(RDY/RDYYR)+(RHR/RHRYR)+(RMIN/RMINYR)
C  SAVE LAT, LONG, AND NO. OF LEVELS PER PROFILE
 77   WRITE(24'IS,ERR=9088)F(1),F(2),NLEV
 78   CONTINUE
C*************************************************************************
C
C*************************************************************************
C   WRITE EDITED MOODS OUTPUT FILE
C   SKIP WRITE ON FIRST PASS IF MISPLACED OR DUP. PROFILE CHECK IS ON
       IF(IMIS.EQ.0.OR.IDUP.EQ.0)GO TO 3
       CALL WEMOOD(D,T,S,NLEV,IEOF)
C
       IF(IEOF.EQ.2)GO TO 9004
C
       GO TO 2
C*************************************************************************
 9000  IF(IPASS.EQ.1)WRITE(6,*)'ERROR IN READING INPUT MOODS DATA '
 900   IF(IPASS.EQ.2)GO TO 98
C
C   FINALIZE STATS AND GET PARAMETERS FOR MISPLACED PROF AT END OF PASS 1
       IF(IMIS.EQ.0)THEN
        ISF=1
        CALL STATS(ISF,DAT,VMEAN,VSDEV,NNUM)
C   SET LIMITS FOR STANDARD DEVIATION CHECK
        SSTMIN=VMEAN-STOL*VSDEV
        SSTMAX=VMEAN+STOL*VSDEV
        REWIND 26
        REWIND 27
C   READ FIRST BAD PROFILE NO.
        READ(26,*,END=90,ERR=90)IBADNO
        GO TO 95
C   SET FLAG TO SIGNIFY NO BAD PROFILES
 90     IBEOF=1
       END IF
C
C*************************************************************************
C
C   FIND DUPLICATE PROFILES ON END OF PASS 1
 95    IF(IDUP.EQ.0)THEN
C   SORT PROFILES IN TIME
        CALL SORT(TIME,INO,KK)
```

```
        DELTIM=IMIN/RMINYR
        CALL DUPGET(TIME,INO,KK,DELTIM,DIST,IDUPER,IDUPCT)
        REWIND 26
        REWIND 27
C   READ FIRST BAD PROFILE NO.
        READ(26,*,END=96,ERR=96)IBADNO
        GO TO 97
C   SET FLAG TO SIGNIFY NO BAD PROFILES
 96     IBEOF=1
       END IF
C
C   MAKE SECOND PASS IF DUP OR MISPL. PROF CHECK IS ON
 97    IF(IMIS.EQ.0.OR.IDUP.EQ.0)THEN
        IPASS=2
        GO TO 3
       END IF
C*******************************************************************************
C
C*******************************************************************************
 98    WRITE(6,*)' '
       WRITE(6,*)'EDIT PARAMETERS'
       WRITE(6,*)'MIN TEMPERATURE (DEG C)           ',TMIN
       WRITE(6,*)'MAX TEMPERATURE (DEG C)           ',TMAX
       WRITE(6,*)'MIN SALINITY                      ',SMIN
       WRITE(6,*)'MAX SALINITY                      ',SMAX
       WRITE(6,*)
       WRITE(6,*)'TEMP/SAL LIMITS PAST DEPTH (M)    ',ZMIN
       WRITE(6,*)'MIN TEMPERATURE (DEG C)           ',TLO1
       WRITE(6,*)'MAX TEMPERATURE (DEG C)           ',THI1
       WRITE(6,*)'MIN SALINITY                      ',SLO1
       WRITE(6,*)'MAX SALINITY                      ',SHI1
       WRITE(6,*)
       WRITE(6,*)'MAX DEPTH (M)                     ',ZMAX
       WRITE(6,*)'BOTTOM DEPTH TOLERANCE            ',BTOL,'%'
       WRITE(6,*)'MAX DENSITY INVERSION  (GM/CM**3) ',DTOL
       WRITE(6,*)'MAX TEMP. INVERSION (DEG C)       ',TTOL
       WRITE(6,*)'INVERSIONS CHECKED AT DEPTHS GREATER THAN ',ZIVC,' M'
       IF(IMIS.EQ.0)WRITE(6,*)'MISPL. PROF. CHECK: MEAN SST (DEG C) ',
     & VMEAN,'  ST. DEV.= ',VSDEV
       IF(IDUP.EQ.0)WRITE(6,*)'DUP. PROF. CHECK: TIME DIFF (MIN) '
     & ,IMIN,'  DIST. DIFF (KM) ',DIST
       WRITE(6,*)
C
       WRITE(29,*)' '
       WRITE(29,*)'EDIT PARAMETERS'
       WRITE(29,*)'MIN TEMPERATURE (DEG C)          ',TMIN
       WRITE(29,*)'MAX TEMPERATURE (DEG C)          ',TMAX
       WRITE(29,*)'MIN SALINITY                     ',SMIN
       WRITE(29,*)'MAX SALINITY                     ',SMAX
       WRITE(29,*)
       WRITE(29,*)'TEMP/SAL LIMITS PAST DEPTH (M)   ',ZMIN
       WRITE(29,*)'MIN TEMPERATURE (DEG C)          ',TLO1
       WRITE(29,*)'MAX TEMPERATURE (DEG C)          ',THI1
       WRITE(29,*)'MIN SALINITY                     ',SLO1
```

```
      WRITE(29,*)'MAX SALINITY                          ',SHI1
      WRITE(29,*)
      WRITE(29,*)'MAX DEPTH (M)                         ',ZMAX
      WRITE(29,*)'BOTTOM DEPTH TOLERANCE                ',BTOL,'%'
      WRITE(29,*)'MAX DENSITY INVERSION  (GM/CM**3) ',DTOL
      WRITE(29,*)'MAX TEMP. INVERSION (DEG C)           ',TTOL
      WRITE(29,*)'INVERSIONS CHECKED AT DEPTHS GREATER THAN ',ZIVC,' M'
      IF(IMIS.EQ.0)WRITE(29,*)'MISPL. PROF. CHECK: MEAN SST (DEG C) ',
     & VMEAN,'  ST. DEV.= ',VSDEV
      IF(IDUP.EQ.0)WRITE(29,*)'DUP. PROF. CHECK: TIME DIFF (MIN) '
     & ,IMIN,'  DIST. DIFF (KM) ',DIST
      WRITE(29,*)
C
      DO 100 I=1,12
 100  IPCT1=IPCT(I)+IPCT1
C
C   SALINITIES ALL ZERO WERE CHANGED TO MISSING, PROF. KEPT
      IPCT1=IPCT1-IPCT(7)
      PC=100.*FLOAT(IPCT1)/FLOAT(ITOTCT)
C
      WRITE(6,*)'TOTAL NO. OF PROFILES PROCESSED  ',ITOTCT
      WRITE(6,*)'NO. OF PROFILES REJECTED         ',IPCT1,' ',PC,'%'
      WRITE(6,*)
      WRITE(6,*)'SUMMARY OF PROFILE REJECTION CAUSES'
C
      WRITE(29,*)'TOTAL NO. OF PROFILES PROCESSED  ',ITOTCT
      WRITE(29,*)'NO. OF PROFILES REJECTED         ',IPCT1,' ',PC,'%'
      WRITE(29,*)
      WRITE(29,*)'SUMMARY OF PROFILE REJECTION CAUSES'
C
      IF(IECK.EQ.0)THEN
       PC=100.*FLOAT(IPCT(1))/FLOAT(ITOTCT)
       WRITE(6,*)'MOODS ERR CODE, DUPLICATES: ',IPCT(1),' ',PC,'%'
       WRITE(29,*)'MOODS ERR CODE, DUPLICATES: ',IPCT(1),' ',PC,'%'
      END IF
      IF(LCHK.EQ.0)THEN
       PC=100.*FLOAT(IPCT(2))/FLOAT(ITOTCT)
       WRITE(6,*)'DATA OVERLAND:              ',IPCT(2),' ',PC,'%'
       WRITE(29,*)'DATA OVERLAND:              ',IPCT(2),' ',PC,'%'
      END IF
       PC=100.*FLOAT(IPCT(3))/FLOAT(ITOTCT)
       WRITE(6,*)'MAX DEPTH EXCEEDED:         ',IPCT(3),' ',PC,'%'
       WRITE(29,*)'MAX DEPTH EXCEEDED:         ',IPCT(3),' ',PC,'%'
      IF(IBCK.EQ.0)THEN
       PC=100.*FLOAT(IPCT(4))/FLOAT(ITOTCT)
       WRITE(6,*)'BOTTOM DEPTH EXCEEDED:      ',IPCT(4),' ',PC,'%'
       WRITE(29,*)'BOTTOM DEPTH EXCEEDED:      ',IPCT(4),' ',PC,'%'
      END IF
      IF(ITBC.EQ.0)THEN
       PC=100.*FLOAT(IPCT(5))/FLOAT(ITOTCT)
       WRITE(6,*)'BAD TEMPERATURE:            ',IPCT(5),' ',PC,'%'
       WRITE(29,*)'BAD TEMPERATURE:            ',IPCT(5),' ',PC,'%'
      END IF
      IF(ISBC.EQ.0)THEN
```

```
        PC=100.*FLOAT(IPCT(6))/FLOAT(ITOTCT)
        WRITE(6,*)'BAD SALINITY:                ',IPCT(6),' ',PC,'%'
        WRITE(29,*)'BAD SALINITY:                ',IPCT(6),' ',PC,'%'
       END IF
       IF(ISOC.EQ.0)THEN
        PC=100.*FLOAT(IPCT(7))/FLOAT(ITOTCT)
        WRITE(6,*)'SALINITIES ALL ZERO:         ',IPCT(7),' ',PC,'%'
        WRITE(29,*)'SALINITIES ALL ZERO:         ',IPCT(7),' ',PC,'%'
       END IF
       IF(ITOC.EQ.0)THEN
        PC=100.*FLOAT(IPCT(8))/FLOAT(ITOTCT)
        WRITE(6,*)'TEMPERATURES ALL ZERO:       ',IPCT(8),' ',PC,'%'
        WRITE(29,*)'TEMPERATURES ALL ZERO:       ',IPCT(8),' ',PC,'%'
       END IF
       IF(IVCK.EQ.0)THEN
        PC=100.*FLOAT(IPCT(9))/FLOAT(ITOTCT)
        WRITE(6,*)'DENSITY INVERSION:           ',IPCT(9),' ',PC,'%'
        WRITE(29,*)'DENSITY INVERSION:           ',IPCT(9),' ',PC,'%'
        PC=100.*FLOAT(IPCT(10))/FLOAT(ITOTCT)
        WRITE(6,*)'TEMPERATURE INVERSION:       ',IPCT(10),' ',PC,'%'
        WRITE(29,*)'TEMPERATURE INVERSION:       ',IPCT(10),' ',PC,'%'
       END IF
C
       IF(IMIS.EQ.0)THEN
        PC=100.*FLOAT(IPCT(11))/FLOAT(ITOTCT)
        WRITE(6,*)'MISPLACED PROFILES:          ',IPCT(11),' ',PC,'%'
        WRITE(29,*)'MISPLACED PROFILES:          ',IPCT(11),' ',PC,'%'
       END IF
C
       IF(IDUP.EQ.0)THEN
        PC=100.*FLOAT(IPCT(12))/FLOAT(ITOTCT)
        WRITE(6,*)'DUPLICATE PROFILES:          ',IPCT(12),' ',PC,'%'
        WRITE(29,*)'DUPLICATE PROFILES:          ',IPCT(12),' ',PC,'%'
       END IF
C
       WRITE(6,*)
       WRITE(6,*)'PROFILE SUMMARY BY SOURCE CODE'
       WRITE(29,*)
       WRITE(29,*)'PROFILE SUMMARY BY SOURCE CODE'
       WRITE(6,*)'    SOURCE CODE','  PROFILES REMOVED',
     & '  PROFILES RETAINED'
       WRITE(29,*)'    SOURCE CODE','  PROFILES REMOVED',
     & '  PROFILES RETAINED'
       DO 102 I=1,23
       WRITE(6,*)I,'    ',ISCODE(I),'      ',ISCODG(I)
 102   WRITE(29,*)I,'    ',ISCODE(I),'      ',ISCODG(I)
C
       WRITE(6,*)'END OF JOB'
       STOP
 9004  WRITE(6,*)'ERROR IN WRITING OUTPUT EDITED MOODS DATA'
       STOP
 9088  WRITE(6,*)'ERROR IN WRITING FILE 24'
       STOP
```

```
 9089  WRITE(6,*)'ERROR IN READING FILE 24'
       STOP
 9090  WRITE(6,*)'ERROR IN READING FILE 26'
       STOP
 9091  WRITE(6,*)'ERROR IN OPENING SCRATCH FILE 26'
       STOP
9092  WRITE(6,*)'ERROR IN OPENING INPUT DATA FILE ',INFILE
       STOP
 9093  WRITE(6,*)'ERROR IN OPENING OUTPUT DATA FILE ',OUTFILE
 9094  WRITE(6,*)'ERROR IN OPENING LOG FILE ',LOGFILE
       END
```

```
      SUBROUTINE DB2DEP(Y,X,D)
C
C          THIS ROUTINE IS AVAILABLE ONLY ON THE UNIVAC.
C          GIVEN LATITUDE (Y) AND LONGITUDE (X), THIS ROUTINE
C          RETURNS DEPTH (D) FROM THE NAVOCEANO DIGITAL BATHY-
C          METRIC DATA BASE ((DB)**2).  COVERAGE INCLUDES BOTH
C          NORTHERN AND SOUTHERN HEMISPHERES FROM APPROXIMATELY
C          81S TO 90N LATITUDE.
C          -- ASCII FORTRAN VERSION --
C
C     PARAMETERS:
C        Y - LATITUDE IN DEGREES (-90 TO 90)
C        X - LONGITUDE IN DEGREES (0 TO 360)
C        D - DEPTH IN METERS
C
        DIMENSION H(4),E(1861),ISQ(2448)
        INTEGER E
C
        DATA(ISQ(J),J=1,195)/
     *   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
     *   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
     *   0,   0,   0,   0,   1,   2,   3,   4,   5,   6,   7,   8,   9,
     *  10,  11,  12,  13,  14,  15,  16,   0,   0,   0,   0,   0,   0,
     *   0,   0,   0,   0,   0,   0,   0,  17,  18,  19,  20,  21,  22,
     *  23,  24,  25,  26,  27,  28,  29,  30,  31,  32,  33,  34,  35,
     *  36,  37,  38,  39,  40,  41,  42,  43,  44,  45,  46,  47,  48,
     *  49,  50,  51,  52,  53,  54,  55,  56,  57,  58,  59,  60,  61,
     *  62,  63,  64,  65,  66,  67,  68,  69,  70,  71,  72,  73,  74,
     *  75,  76,  77,  78,  79,  80,  81,  82,  83,  84,  85,  86,  87,
     *  88,  89,  90,  91,  92,  93,  94,  95,  96,  97,  98,  99, 100,
     * 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113,
     * 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126,
     * 127, 128, 129, 130, 131, 132, 133, 134, 135, 136, 137, 138, 139,
     * 140, 141, 142, 143, 144, 145, 146, 147, 148, 149, 150, 151, 152/
        DATA(ISQ(J),J=196,390)/
     * 153, 154, 155, 156, 157, 158, 159, 160, 161, 162, 163, 164, 165,
     * 166, 167, 168, 169, 170, 171, 172, 173, 174, 175, 176, 177, 178,
     * 179, 180, 181, 182, 183, 184, 185, 186, 187, 188, 189, 190, 191,
     * 192, 193, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204,
     * 205, 206, 207, 208, 209, 210, 211, 212, 213, 214, 215, 216, 217,
     * 218, 219, 220, 221, 222, 223, 224, 225, 226, 227, 228, 229, 230,
     * 231, 232, 233, 234, 235, 236, 237, 238, 239, 240, 241, 242, 243,
     * 244, 245, 246, 247, 248, 249, 250, 251, 252, 253, 254, 255, 256,
     * 257, 258, 259, 260, 261, 262, 263, 264, 265, 266, 267, 268, 269,
     * 270, 271, 272, 273, 274, 275, 276, 277, 278, 279, 280, 281, 282,
     * 283, 284, 285, 286, 287, 288, 289, 290, 291, 292, 293, 294, 295,
     * 296, 297, 298, 299, 300, 301, 302, 303, 304, 305, 306, 307, 308,
     * 309, 310, 311, 312, 313, 314, 315, 316, 317, 318, 319, 320, 321,
     * 322, 323, 324, 325, 326, 327, 328, 329, 330, 331, 332, 333, 334,
     * 335, 336, 337, 338, 339, 340, 341, 342, 343, 344, 345, 346, 347/
        DATA(ISQ(J),J=391,585)/
     * 348, 349, 350, 351, 352, 353, 354, 355, 356, 357, 358, 359, 360,
     * 361, 362, 363, 364, 365, 366, 367, 368, 369, 370, 371, 372, 373,
```

```
     * 374, 375, 376, 377, 378, 379, 380, 381, 382, 383, 384, 385, 386,
     * 387, 388, 389, 390, 391, 392, 393, 394, 395, 396, 397, 398, 399,
     * 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412,
     * 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 425,
     * 426, 427, 428, 429, 430, 431, 432, 433, 434, 435, 436, 437, 438,
     * 439, 440, 441, 442, 443, 444, 445, 446, 447, 448, 449, 450, 451,
     * 452, 453, 454, 455, 456, 457, 458, 459, 460, 461, 462, 463, 464,
     * 465, 466, 467, 468, 469, 470, 471, 472, 473, 474, 475, 476, 477,
     * 478, 479, 480, 481, 482, 483, 484, 485, 486, 487, 488, 489, 490,
     * 491, 492, 493, 494, 495, 496, 497, 498, 499, 500, 501, 502, 503,
     * 504, 505, 506, 507, 508, 509, 510, 511, 512, 513, 514, 515, 516,
     * 517, 518, 519, 520, 521, 522, 523, 524, 525, 526, 527, 528, 529,
     * 530, 531, 532, 533, 534, 535, 536, 537, 538, 539, 540, 541, 542/
      DATA(ISQ(J),J=586,780)/
     * 543, 544, 545, 546, 547, 548, 549, 550, 551, 552, 553, 554, 555,
     * 556, 557, 558, 559, 560, 561, 562, 563, 564, 565, 566, 567, 568,
     * 569, 570, 571, 572, 573, 574, 575, 576, 577, 578, 579, 580, 581,
     * 582, 583, 584, 585, 586, 587, 588, 589, 590, 591, 592, 593, 594,
     * 595, 596, 597, 598, 599, 600, 601, 602, 603, 604, 605, 606, 607,
     * 608, 609, 610, 611, 612, 613, 614, 615, 616, 617, 618, 619, 620,
     * 621, 622, 623, 624, 625, 626, 627, 628, 629, 630, 631, 632, 633,
     * 634, 635, 636, 637, 638, 639, 640, 641, 642, 643, 644, 645, 646,
     * 647, 648, 649, 650, 651, 652, 653, 654, 655, 656, 657, 658, 659,
     * 660, 661, 662, 663, 664, 665, 666, 667, 668, 669, 670, 671, 672,
     * 673, 674, 675, 676, 677, 678, 679, 680, 681, 682, 683, 684, 685,
     * 686, 687, 688, 689, 690, 691, 692, 693, 694, 695, 696, 697, 698,
     * 699, 700, 701, 702, 703, 704, 705, 706, 707, 708, 709, 710, 711,
     * 712, 713, 714, 715, 716, 717, 718, 719, 720, 721, 722, 723, 724,
     * 725, 726, 727, 728, 729, 730, 731, 732, 733, 734, 735, 736, 737/
      DATA(ISQ(J),J=781,975)/
     * 738, 739, 740, 741, 742, 743, 744, 745, 746, 747, 748, 749, 750,
     * 751, 752, 753,   0,   0, 754, 755, 756, 757, 758, 759, 760, 761,
     * 762, 763, 764, 765, 766, 767, 768, 769, 770, 771, 772, 773, 774,
     * 775, 776, 777, 778, 779, 780, 781, 782, 783, 784, 785, 786, 787,
     * 788, 789, 790, 791, 792, 793, 794, 795, 796, 797, 798, 799, 800,
     * 801, 802, 803, 804, 805, 806, 807, 808, 809, 810, 811, 812, 813,
     * 814, 815, 816, 817, 818, 819, 820, 821, 822, 823,   0,   0, 824,
     * 825, 826, 827, 828, 829, 830, 831, 832, 833, 834, 835, 836, 837,
     * 838, 839, 840, 841, 842, 843, 844, 845, 846, 847, 848, 849, 850,
     * 851, 852, 853, 854, 855, 856, 857, 858, 859, 860, 861, 862, 863,
     * 864, 865, 866, 867, 868, 869, 870, 871, 872, 873, 874, 875, 876,
     * 877, 878, 879, 880, 881, 882, 883, 884, 885, 886, 887, 888, 889,
     * 890, 891, 892, 893,   0,   0, 894, 895, 896, 897, 898, 899, 900,
     * 901, 902, 903, 904, 905, 906, 907, 908, 909, 910, 911, 912, 913,
     * 914, 915, 916, 917, 918, 919, 920, 921, 922, 923, 924, 925, 926/
      DATA(ISQ(J),J=976,1170)/
     * 927, 928, 929, 930, 931, 932, 933, 934, 935, 936, 937, 938, 939,
     * 940, 941, 942, 943, 944, 945, 946, 947, 948, 949, 950, 951, 952,
     * 953, 954, 955, 956, 957, 958, 959, 960, 961, 962, 963,   0,   0,
     * 964, 965, 966, 967, 968, 969, 970, 971, 972, 973, 974, 975, 976,
     * 977, 978, 979, 980, 981, 982, 983, 984, 985, 986, 987, 988, 989,
     * 990, 991, 992, 993, 994, 995, 996, 997, 998, 999,1000,1001,1002,
     *1003,1004,1005,1006,1007,1008,1009,1010,1011,1012,1013,1014,1015,
     *1016,1017,1018,1019,1020,1021,1022,1023,1024,1025,1026,1027,1028,
```

```
*1029,1030,1031,1032,1033,   0,   0,1034,1035,1036,1037,1038,1039,
*1040,1041,1042,1043,1044,1045,1046,1047,1048,1049,1050,1051,1052,
*1053,1054,1055,1056,1057,1058,1059,1060,1061,1062,1063,1064,1065,
*1066,1067,1068,1069,1070,1071,1072,1073,1074,1075,1076,1077,1078,
*1079,1080,1081,1082,1083,1084,1085,1086,1087,1088,1089,1090,1091,
*1092,1093,1094,1095,1096,1097,1098,1099,1486,1487,1488,1489,   0,
*   0,1100,1101,1102,1103,1104,1105,1106,1107,1108,1109,1160,1161/
 DATA(ISQ(J),J=1171,1365)/
*1162,1163,1164,1165,1166,1167,1208,1209,1210,1211,1212,1213,1214,
*1215,1256,1257,1258,1259,1260,1261,1262,1263,1304,1305,1306,1307,
*1308,1309,1310,1311,1352,1353,1354,1355,1356,1382,1383,1384,1385,
*1386,1387,1388,1389,1390,1436,1437,1438,1439,1440,1441,1442,1443,
*1484,1485,1492,1493,1494,1495,   0,   0,1110,1111,1112,1113,1114,
*1115,1116,1117,1118,1119,1168,1169,1170,1171,1172,1173,1174,1175,
*1216,1217,1218,1219,1220,1221,1222,1223,1264,1265,1266,1267,1268,
*1269,1270,1271,1312,1313,1314,1315,1316,1317,1318,1319,1357,1358,
*1359,1360,1361,1391,1392,1393,1394,1395,1396,1397,1398,1399,1444,
*1445,1446,1447,1448,1449,1450,1451,1490,1491,1498,1499,1500,1501,
*   0,   0,1120,1121,1122,1123,1124,1125,1126,1127,1128,1129,1176,
*1177,1178,1179,1180,1181,1182,1183,1224,1225,1226,1227,1228,1229,
*1230,1231,1272,1273,1274,1275,1276,1277,1278,1279,1320,1321,1322,
*1323,1324,1325,1326,1327,1362,1363,1364,1365,1366,1400,1401,1402,
*1403,1404,1405,1406,1407,1408,1452,1453,1454,1455,1456,1457,1458/
 DATA(ISQ(J),J=1366,1560)/
*1459,1496,1497,1504,1505,1506,1507,   0,   0,1130,1131,1132,1133,
*1134,1135,1136,1137,1138,1139,1184,1185,1186,1187,1188,1189,1190,
*1191,1232,1233,1234,1235,1236,1237,1238,1239,1280,1281,1282,1283,
*1284,1285,1286,1287,1328,1329,1330,1331,1332,1333,1334,1335,1367,
*1368,1369,1370,1371,1409,1410,1411,1412,1413,1414,1415,1416,1417,
*1460,1461,1462,1463,1464,1465,1466,1467,1502,1503,1510,1511,1512,
*1513,   0,   0,1140,1141,1142,1143,1144,1145,1146,1147,1148,1149,
*1192,1193,1194,1195,1196,1197,1198,1199,1240,1241,1242,1243,1244,
*1245,1246,1247,1288,1289,1290,1291,1292,1293,1294,1295,1336,1337,
*1338,1339,1340,1341,1342,1343,1372,1373,1374,1375,1376,1418,1419,
*1420,1421,1422,1423,1424,1425,1426,1468,1469,1470,1471,1472,1473,
*1474,1475,1508,1509,1516,1517,1518,1519,1716,1717,1150,1151,1152,
*1153,1154,1155,1156,1157,1158,1159,1200,1201,1202,1203,1204,1205,
*1206,1207,1248,1249,1250,1251,1252,1253,1254,1255,1296,1297,1298,
*1299,1300,1301,1302,1303,1344,1345,1346,1347,1348,1349,1350,1351/
 DATA(ISQ(J),J=1561,1755)/
*1377,1378,1379,1380,1381,1427,1428,1429,1430,1431,1432,1433,1434,
*1435,1476,1477,1478,1479,1480,1481,1482,1483,1514,1515,1718,1719,
*1720,1721,1722,1723,1724,1725,1750,1751,1752,1753,1754,1755,   0,
*   0,   0,   0,   0,   0,   0,   0,   0,1520,1521,1522,1532,1533,
*1534,1535,1536,1537,1538,1539,1564,1565,1566,1567,1568,1569,1570,
*1571,1596,1597,1598,1599,1600,1601,1602,1603,1628,1629,1630,1631,
*1632,1633,1652,1653,1654,1655,1656,1657,1658,1659,1684,1685,1686,
*1687,1688,1689,1690,1691,1726,1727,1728,1729,1730,1731,1732,1733,
*1756,1757,1758,1759,1760,1761,   0,   0,   0,   0,   0,   0,   0,
*   0,   0,1523,1524,1525,1540,1541,1542,1543,1544,1545,1546,1547,
*1572,1573,1574,1575,1576,1577,1578,1579,1604,1605,1606,1607,1608,
*1609,1610,1611,1634,1635,1636,1637,1638,1639,1660,1661,1662,1663,
*1664,1665,1666,1667,1692,1693,1694,1695,1696,1697,1698,1699,1734,
*1735,1736,1737,1738,1739,1740,1741,1762,1763,1764,1765,1766,1767,
```

```
*   0,   0,   0,   0,   0,   0,   0,   0,   0,1526,1527,1528,1548/
 DATA(ISQ(J),J=1756,1950)/
*1549,1550,1551,1552,1553,1554,1555,1580,1581,1582,1583,1584,1585,
*1586,1587,1612,1613,1614,1615,1616,1617,1618,1619,1640,1641,1642,
*1643,1644,1645,1668,1669,1670,1671,1672,1673,1674,1675,1700,1701,
*1702,1703,1704,1705,1706,1707,1742,1743,1744,1745,1746,1747,1748,
*1749,1768,1769,1770,1771,1772,1773,   0,   0,   0,   0,   0,   0,
*   0,   0,   0,1529,1530,1531,1556,1557,1558,1559,1560,1561,1562,
*1563,1588,1589,1590,1591,1592,1593,1594,1595,1620,1621,1622,1623,
*1624,1625,1626,1627,1646,1647,1648,1649,1650,1651,1676,1677,1678,
*1679,1680,1681,1682,1683,1708,1709,1710,1711,1712,1713,1714,1715,
*1894,1895,1912,1913,1914,1915,1916,1917,   0,   0,   0,   0,   0,
*   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,1774,1775,
*1776,1777,1778,1779,1780,1781,1798,1799,1800,1801,1802,1803,1804,
*1805,1822,1823,1824,1825,1826,1827,1828,1829,   0,   0,   0,   0,
*1846,1847,1848,1849,1850,1851,1864,1865,1866,1867,1868,1869,1870,
*1871,1888,1889,1890,1891,1892,1893,1902,1903,1918,1919,1920,1921/
 DATA(ISQ(J),J=1951,2145)/


*1922,1923,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
*   0,   0,   0,   0,   0,1782,1783,1784,1785,1786,1787,1788,1789,
*1806,1807,1808,1809,1810,1811,1812,1813,1830,1831,1832,1833,1834,
*1835,1836,1837,   0,   0,   0,   0,1852,1853,1854,1855,1856,1857,
*1872,1873,1874,1875,1876,1877,1878,1879,1896,1897,1898,1899,1900,
*1901,1910,1911,1924,1925,1926,1927,1928,1929,2030,2031,2032,2033,
*2034,2035,2036,2037,2054,2055,2056,2057,2058,2059,2060,2061,1790,
*1791,1792,1793,1794,1795,1796,1797,1814,1815,1816,1817,1818,1819,
*1820,1821,1838,1839,1840,1841,1842,1843,1844,1845,1962,1963,1964,
*1965,1858,1859,1860,1861,1862,1863,1880,1881,1882,1883,1884,1885,
*1886,1887,1904,1905,1906,1907,1908,1909,2014,2015,2016,2017,2018,
*2019,2020,2021,2038,2039,2040,2041,2042,2043,2044,2045,2062,2063,
*2064,2065,2066,2067,2068,2069,2078,2079,2080,2081,2082,2083,2084,
*2085,1930,1931,1932,1933,1934,1935,1936,1937,1946,1947,1948,1949,
*1950,1951,1952,1953,1966,1967,1968,1969,1970,1971,1972,1973,1982/
 DATA(ISQ(J),J=2146,2340)/
*1983,1984,1985,1986,1987,1988,1989,1998,1999,2000,2001,2002,2003,
*2004,2005,2022,2023,2024,2025,2026,2027,2028,2029,2046,2047,2048,
*2049,2050,2051,2052,2053,2070,2071,2072,2073,2074,2075,2076,2077,
*2086,2087,2088,2089,2090,2091,2092,2093,1938,1939,1940,1941,1942,
*1943,1944,1945,1954,1955,1956,1957,1958,1959,1960,1961,1974,1975,
*1976,1977,1978,1979,1980,1981,1990,1991,1992,1993,1994,1995,1996,
*1997,2006,2007,2008,2009,2010,2011,2012,2013,2094,2095,2096,2097,
*2098,2099,2100,2101,2102,2103,2104,2105,2106,2107,2108,2109,2110,
*2111,2148,2149,2150,2151,2152,2153,2154,2155,2156,2157,2158,2159,
*2160,2161,2162,2163,2164,2165,2202,2203,2204,2205,2206,2207,2208,
*2209,2210,2211,2212,2213,2214,2215,2216,2217,2218,2219,2256,2257,
*2258,2259,2260,2261,2262,2263,2264,2265,2266,2267,2268,2269,2270,
*2271,2272,2273,2112,2113,2114,2115,2116,2117,2118,2119,2120,2121,
*2122,2123,2124,2125,2126,2127,2128,2129,2166,2167,2168,2169,2170,
*2171,2172,2173,2174,2175,2176,2177,2178,2179,2180,2181,2182,2183/
 DATA(ISQ(J),J=2341,2448)/
*2220,2221,2222,2223,2224,2225,2226,2227,2228,2229,2230,2231,2232,
*2233,2234,2235,2236,2237,2274,2275,2276,2277,2278,2279,2280,2281,
```

```
     *2282,2283,2284,2285,2286,2287,2288,2289,2290,2291,2130,2131,2132,
     *2133,2134,2135,2136,2137,2138,2139,2140,2141,2142,2143,2144,2145,
     *2146,2147,2184,2185,2186,2187,2188,2189,2190,2191,2192,2193,2194,
     *2195,2196,2197,2198,2199,2200,2201,2238,2239,2240,2241,2242,2243,
     *2244,2245,2246,2247,2248,2249,2250,2251,2252,2253,2254,2255,2292,
     *2293,2294,2295,2296,2297,2298,2299,2300,2301,2302,2303,2304,2305,
     *2306,2307,2308,2309/
C
      DATA IOP/0/
C
      IF(IOP.EQ.0)OPEN(UNIT=22,FILE='BATHY*DBDB',ACCESS='DIRECT',
     &FORM='UNFORMATTED',STATUS='OLD',ERR=9090,RECL=1865)
      IOP=1
C
      D=0.
      XX=X
      IF(X.EQ.360.)XX=0.
      C=5./60.
      C2=C/2.
C
      IS=(Y+80.)/5.
      ISN=IS*72+XX/5.+1
C
      IF(ISN.EQ.ISNO)GO TO 4
      ISNO=ISN
      IF(ISN.LT.1)GO TO 9
      IF(ISN.GT.2448)GO TO 9
      IRP=ISQ(ISN)
      IF(IRP.EQ.0)GO TO 9
C
      READ(22'IRP,ERR=9)H,E
C
      Y2=H(3)-5.
      X2=H(4)+5.
      IF(XX.LT.H(4).OR.XX.GT.X2)GO TO 8
C
    4 IF(IRP.EQ.0)GO TO 9
      I=(XX-H(4)+C2)/C+1
      J=(H(3)-Y+C2)/C+1
      K=(I-1)*61+J
      L=(K+1)/2
C
      IF(MOD(K,2).EQ.0)GO TO 6
      D=BITS(E(L),1,18)
      RETURN
    6 D=BITS(E(L),19,18)
      RETURN
    8 WRITE(6,100)Y,X,Y2,H(3),H(4),X2
  100 FORMAT(1X,' WRONG SQUARE -- ASKED FOR:  Y = ',F9.3,'  X = ',
     * F9.3,'    GOT:  Y1 = ',F9.3,'  Y2 = ',F9.3,'  X1 = ',F9.3,
     * '  X2 = ',F9.3)
      RETURN
    9 CONTINUE
C   9 WRITE(6,200)ISN,Y,X,IRP
```

```
C 200 FORMAT(1X,' ISN = ',I5,'  NO DATA FOR Y = ',F9.3,'  X = ',F9.3,/
C     * 1X,' IRP = ',I5)
      RETURN
 9090 WRITE(6,*)'ERROR IN OPENING UNIT 22, BATHY*DBDB'
      STOP
      END
```

```
      SUBROUTINE DEPTHC(D,T,S,NLEV,IZFLG,IDFLG)
C
C   THIS PROGRAM CHECKS FOR NEGATIVE DEPTHS (IDFLG=1),
C   FOR NON-MONOTONIC DEPTHS (IDFLG=2), AND FOR DEPTHS
C   EXCEEDING A MAX VALUE (IZFLG=1).
C   NEGATIVE DEPTHS ARE MADE POSITIVE.
C   IF DEPTHS WERE NEGATIVE AND ARE NON-MONOTONIC WHEN
C   MADE POSITIVE, IDFLG=3.
C   DUPLICATE DEPTHS ARE REMOVED, AND THE DEPTHS ARE
C   THEN SORTED.
C
      COMMON /ZLIM/ ZMAX,ZVAL
      DIMENSION D(1),T(1),S(1)
C
      IDFLG=0
      IZFLG=0
C
C   IF NO DATA RETURN
      IF(NLEV.EQ.0)RETURN
C
C   CHECK FOR NEGATIVE DEPTHS.
      DO 100 I=1,NLEV
      IF(D(I).LT.0.0)THEN
       IDFLG=1
       D(I)=-D(I)
      END IF
C   CHECK FOR DEPTHS EXCEEDING MAX VALUE
      IF(D(I).GT.ZMAX)THEN
       IZFLG=1
       ZVAL=D(I)
       RETURN
      END IF
  100 CONTINUE
C
C
C   SORT DEPTH LEVELS
      IF(NLEV.LE.1)GO TO 999
      DO 150 I=1,NLEV
      IFLAG=0
      DO 140 J=1,NLEV-1
      IF(D(J).GT.D(J+1))THEN
       XD=D(J+1)
       XT=T(J+1)
       XS=S(J+1)
       D(J+1)=D(J)
       D(J)=XD
       T(J+1)=T(J)
       T(J)=XT
       S(J+1)=S(J)
       S(J)=XS
       IFLAG=1
       IF(IDFLG.EQ.0)IDFLG=2
       IF(IDFLG.EQ.1)IDFLG=3
      END IF
```

```
  140 CONTINUE
      IF(IFLAG.EQ.0)GO TO 180
  150 CONTINUE
C
C
C  ELIMINATE DUPLICATIONS
 180  DO 200 J=1,NLEV-1
      IF(D(J+1)-D(J).LT.0.01)D(J)=-1.
  200 CONTINUE
      ICT=0
      DO 250 J=1,NLEV
      IF(D(J).LT.0)GO TO 250
      ICT=ICT+1
      D(ICT)=D(J)
      T(ICT)=T(J)
      S(ICT)=S(J)
  250 CONTINUE
      NLEV=ICT
C
 999   RETURN
       END
```

```
      SUBROUTINE DUPGET(TIME,INO,KK,DELTIM,DELDIS,IDUPER,ICT)
CC
C   THIS PROGRAM CHECKS TIME-SORTED PROFILE PAIRS FOR DUPLICATES.
C   WHEN PROFILES ARE DUPLICATED, THAT IS, THEY FALL IN THE SAME
C   TIME (DELTIM) - DISTANCE (DELDIS) WINDOW, THE SHORTER PROFILE(S)
C   IS ELIMINATED.  IF THEY ARE THE SAME LENGTH, THE FIRST PROFILE
C   IS KEPT.  DISTANCES ARE CHECKED FOR PROFILES IN THE SAME TIME
C   WINDOW BETWEEN THE FIRST PROFILE AND EACH OF THE OTHER
C   PROFILES.  IF THE FIRST PROFILE IN THE WINDOW IS FLAGGED
C   AS A DUPLICATE, THE WINDOW BEGINS AGAIN AT THE FIRST PROFILE
C   NOT FLAGGED AS DUPLICATE.  OTHERWISE, ALL PROFILE DISTANCES
C   FROM THE FIRST PROFILE ARE CHECKED AND DUPLICATES ARE
C   FLAGGED.  THE NEXT TIME WINDOW BEGINS AT THE FIRST NON-
C   FLAGGED PROFILE.  DUPLICATE PROFILE NUMBERS ARE STORED IN
C   IDUPER.
CC
      DIMENSION INO(1),TIME(1),IDUPER(1)
C      EQUIVALENCE (TIME(1),IDUPER(1))
C
      DO 10 I=2,KK
C
C   SKIP DUPLICATES ALREADY FLAGGED
      IF(INO(I-1).LT.0)GO TO 10
      I1=I
 2    CONTINUE
     IF(INO(I1).LT.0)THEN
       I1=I1+1
       IF(I1.GT.KK)GO TO 10
       GO TO 2
      END IF
C
C   CHECK TIME WINDOW
CC
     IF(TIME(I1)-TIME(I-1).LE.DELTIM)THEN
CC
C   CHECK DISTANCE WINDOW
C   RETRIEVE LAT, LONG, AND NO. OF LEVELS
      READ(24'INO(I-1),ERR=9010)ALAT1,ALON1,NLEV1
 5    READ(24'INO(I1),ERR=9010)ALAT,ALON,NLEV
C   GET DISTANCE IN KM
      CALL GMLIC(ALAT,ALON,ALAT1,ALON1,DIS)
C   IF DUPLICATE IS FOUND, SET FLAG - MAKE PROFILE NO. NEG.
      IF(DIS.LT.DELDIS)THEN
       IDFL=1
       IF(NLEV.GT.NLEV1)THEN
        IN=INO(I1)
        IOUT=INO(I-1)
        INO(I-1)=-INO(I-1)
       END IF
       IF(NLEV.LE.NLEV1)THEN
        IN=INO(I-1)
        IOUT=INO(I1)
        INO(I1)=-INO(I1)
       END IF
```

```
        WRITE(29,*)'DUPLICATE PROFILES ',IN,' - ',IOUT,' REMOVE ',IOUT
       END IF
C   START NEXT TIME WINDOW IF PRESENT WINDOW BEGINS WITH DUP.
       IF(INO(I-1).LT.0)GO TO 10
C   CHECK WHETHER NEXT PROFILE LIES IN THE TIME WINDOW
  8    I1=I1+1
       IF(I1.GT.KK)GO TO 10
C   SKIP PROFILES ALREADY FLAGGED
       IF(INO(I1).LT.0)GO TO 8
       IF(TIME(I1)-TIME(I-1).LE.DELTIM)GO TO 5
CC
      END IF
CC
C
 10   CONTINUE
C
C
C   IF NO DUPLICATES ARE FOUND, RETURN
      ICT=0
      IF(IDFL.EQ.0)RETURN
C   STORE NEGATIVE INO'S (AS POSITIVE) IN IDUPER, IDUPER CONTAINS
C   PROFILE NUMBERS OF DUPLICATE PROFILES.  IDUPER WAS NOT FILLED
C   DIRECTLY IN ABOVE SECTION IN ORDER TO CONSERVE ARRAY SPACE.
      DO 20 I=1,KK
       IF(INO(I).LT.0)THEN
        ICT=ICT+1
        IDUPER(ICT)=ABS(INO(I))
       END IF
 20   CONTINUE
C
C   SORT IDUPER
      IF(ICT.LE.1)RETURN
      DO 40 I=1,ICT
       IFLAG=0
        DO 30 J=1,ICT-1
         IF(IDUPER(J).GT.IDUPER(J+1))THEN
          IZ=IDUPER(J+1)
          IDUPER(J+1)=IDUPER(J)
          IDUPER(J)=IZ
          IFLAG=1
         END IF
 30      CONTINUE
       IF(IFLAG.EQ.0)RETURN
 40   CONTINUE
C
      RETURN
 9010 WRITE(6,*)' ERROR IN READING UNIT 24'
      STOP
      END
```

```
      SUBROUTINE GMLIC (ALATA,ALONA,ALATB,ALONB,S)
C
C   THIS ROUTINE COMPUTES DISTANCE, S, BETWEEN ALATA,ALONA
C   AND ALATB,ALONB.
C
      DATA ARC1/0.484813681110E-5/,AXIS/6378206.4/,
     1ESQ/0.676865799729E-2/
C
C        STATEMENT FUNCTIONS
C
C      R1(D)=.77319978E-5*(1258.0483-D*(68.112835+D))
       R1(D)=.77319978E-5*(1258.0483-D*(68.112835+D))
       R2(D)=.97935127E-2*D
       S1(D)=.95512914E-5*(20.819968-D*(40.782792+D))
       S2(D)=.13348137E-3*(147.73977+D*(71.369887+D))
       T1(D)=.909646805E-6*(32444.5815-D*(144.739770+D))
       T2(D)=.13348137E-3*(147.73977-D*(148.73977-D))
C
C   CONVERT LAT AND LONG FROM DEGREES TO SECONDS
       ALAT1=ALATA*3600.
       ALON1=ALONA*3600.
       ALAT2=ALATB*3600.
       ALON2=ALONB*3600.
C
C        COMPUTATION OF DELTA LAMDA
C
       XPHIM=((ALAT1+ALAT2)/2.)*ARC1
       XSIN=SIN (XPHIM)
       XCOS=COS (XPHIM)
       D=XSIN**2
       DLONG=ALON2-ALON1
       DLAT=ALAT2-ALAT1
       V=(DLAT/100000.)**2
       W=(DLONG/100000.)**2
       AN=SQRT (1.-ESQ*D)
       XRACV=AXIS/AN
       XRACM=(AXIS*(1.-ESQ))/(AN*(1.-ESQ*D))
       XH=XCOS*XRACV*ARC1
       XB=XRACM*ARC1
       XIK=DLONG*XH*(1.+V*R1(D)-W*R2(D))
       YIK=DLAT*XB*(1.+V*S1(D)-W*S2(D))
       DELLA =  .5*((DLONG*XSIN)*(1. + V*T1(D) + W*T2(D)))
       S=SQRT (XIK**2+YIK**2)
C   DIST IN KM
       S=S/1000.
       RETURN
       END
```

```
      SUBROUTINE INVERC(D,T,S,NLEV,ZIVC,INVFLG)
C  THIS PROGRAM CHECKS FOR DENSITY OR TEMPERATURE
C  INVERSIONS ACCORDING TO THE TOLERANCES DENTOL
C  AND TTOL.
C
      COMMON/INVERT/DENTOL,TTOL,BTOL,DELDEN,DELT
      DIMENSION D(1),T(1),S(1),INVFLG(1)
C
      IC=0
      INVFLG(1)=0
      INVFLG(2)=0
C  CHECK FOR TEMPERATURE INVERSIONS
      IC=0
      DO 500 I=1,NLEV
      IF(D(I).LT.ZIVC)GO TO 200
      IF(T(I).LT.-800.)GO TO 500
C
      IF(IC.EQ.0)THEN
       IC=1
       T1=T(I)
       GO TO 500
      END IF
C
      DELT=T(I)-T1
      IF(DELT.GT.TTOL)THEN
       INVFLG(2)=1
       GO TO 510
      END IF
C
C  LOCAL MIN AND MAX TEMP IN INVERSION ARE USED
      TSAV=T1
      IF(T(I).GT.TSAV)THEN
       T1=TSAV
      ELSE
       T1=T(I)
      END IF
  500 CONTINUE
  510 CONTINUE
C
C  CHECK FOR DENSITY INVERSIONS
      IC=0
      DO 200 I=1,NLEV
      IF(D(I).LT.ZIVC)GO TO 200
      IF(T(I).LT.-800.0.OR.S(I).LT.-800.0)GO TO 200
      IF(S(I).LT.0.001)GO TO 200
      DEN=EQSTAT(T(I),S(I),D(I))
C
      IF(IC.EQ.0)THEN
       IC=1
       DEN1=DEN
       GO TO 100
      END IF
C
      DELDEN=DEN1-DEN
```

```
      IF(DELDEN.GT.DENTOL)THEN
       INVFLG(1)=1
       GO TO 999
      END IF
C
C   LOCAL MIN AND MAX DENSITY IN INVERSION ARE USED
      DSAV=DEN1
      IF(DEN.LT.DSAV)THEN
        DEN1=DSAV
       ELSE
        DEN1=DEN
      END IF
C
 100  CONTINUE
 200  CONTINUE
C
 999  RETURN
      END
C
C
C EQUATION OF STATE OF SEA WATER- CHEN & MILLERO
C DEEP SEA RESEARCH, 1977,VOL 24,PP 365-369
          FUNCTION EQSTAT(T,S,P)
C ALL UNITS CGS
C T IN DEGREES C
C S IN PARTS PER THOUSAND
C P IN DECIBARS (1DB ABOUT EQUALS 1M)
C DENSITY IN GR/CM**3
        DO=1.0281045-5.35633E-05*T-6.78195E-06*T*T
     *  +7.0517E-08*T**3-8.4794E-10*T**4+5.057E-12*T**5
     *  +(8.0792E-04-3.2481E-06*T+6.423E-08*T**2-6.490E-10*T**3)
     *  *(S-35.)+2.045E-07*(S-35.)**2
              VO=1.0/DO
          KO=21585.72+132.5657*T-2.0860*T*T+8.7648E-03*T**3
     *    +(56.928-0.2975*T)*(S-35.)
                A=3.40075-7.6371E-3*T+2.9651E-4*T**2
     *  +(2.287E-3-3.255E-4*T)*(S-35.)
                B=2.211E-05
          PP=0.1*P
            VP=VO-VO*PP/(KO+A*PP+B*PP**2)
          DENS=1.0/VP
          EQSTAT=DENS
          RETURN
          END
```

```
C       SUBROUTINE LANDMASK READS DATA FROM THE FILE,LANDMASK.DAT
C       AND RETURNS A VALUE OF 1 OR 0 DEPENDING ON WHETHER THE WHETHER THE
C       DATA IN THE SPECIFIED AREA IS LAND OR WATER.
C       THE INPUT PARAMETERS ARE:
C               DLATD = DEGREE LATITUDE REQUESTED
C               DLATM = MINUTE LATITUDE REQUESTED
C               DLOND = DEGREE LONGITUDE REQUESTED
C               LONM  = MINUTE LONGITUDE REQUESTED
C       THE OUTPUT PARAMETER IS:
C               IMASK = AN INTEGER WITH THE VALUE OF 1 IF THE
C                       REQUESTED RECORD IS LAND OR 0 IF THE
C                       REQUESTED RECORD IS WATER
C       SUBROUTINE LANDMASK WAS WRITTEN BY J. HAMMACK AND
C       E. GREMILLION, NAVOCEANO, CODE 8321.  THE LANDMASK
C       DATA FILE WAS BASED ON WORK DONE BY MIKE CARRON, CODE 022.
        SUBROUTINE LANDMASK(DLATD,DLATM,DLOND,LONM,MASK)
        IMPLICIT INTEGER (A-Z)
        DIMENSION OUT(5)
        DATA ICK/1/
        LATD=DLATD
        LATM=DLATM
        LOND=DLOND
        IF (PASS.EQ.0)THEN
          OPEN (UNIT=10,FILE='DRBO:[BILL.MOODS]LANDMASK.DAT',
       1    ACCESS='DIRECT',FORM='UNFORMATTED',RECL=5,
       2    MAXREC=52200,STATUS='OLD',READONLY)
        END IF
        PASS=1
        IF (LATD.LE.0.AND.LATM.LT.0) THEN
          LATD = LATD - 1
          LATM = ABS(LATM)
        END IF

        IF (DLOND.LT.0) LOND = DLOND +360
        IF (LONM.LT.0) LONM = ABS(LONM)
        RECORD = (LATD + 72)*360 + LOND + 1
        IF(ICK.EQ.1)GO TO 10
        IF(RECORD.EQ.RECSAV)RETURN
 10     READ (10'RECORD) OUT
        K = (LATM/5)*12 + (LONM/5)
        MASK = IBITS(OUT(1),K,1)
        ICK=0
        RECSAV=RECORD
        RETURN
        END
```

```
      COMPILER (LINK=IBJ$)
C
C     THIS ROUTINE IS AVAILABLE ONLY ON THE UNIVAC.
C     SUBROUTINE TO CHECK FOR LAND AT TEN MINUTE GRID INTERVAL
C
C     WRITTEN BY J. C. DEPNER
C     DATE : 06-28-84
C
C           X1 = LATITUDE DEGREES (OR DEGREES DECIMAL)
C           X2 = LATITUDE MINUTES
C           X3 = LONGITUDE DEGREES (OR DEGREES DECIMAL)
C           X4 = LONGITUDE MINUTES
C           X5 = 0 FOR WATER , 1 FOR LAND
C
      SUBROUTINE LNDCHK (X1,X2,X3,X4,X5)
      INTEGER ARRAY(27)
      IF(IFIRST.EQ.0)THEN
C
C       ASSIGN MAP FILE.
C
        OPEN(28,FILE='CIA*ASCII',STATUS='OLD',ACCESS='DIRECT',
     *  FORM='UNFORMATTED',RECL=27,RCDS=64389,ASSOC=IKEY)
        IFIRST=1
      END IF
C
C     ADD 90 TO LATITUDE
C     ADD 360 TO LONGITUDE IF NEGATIVE
C
      XLAT=(X1+(X2/60.0)*SIGN(1.0,X1))+90.0
      XLON=(X3+(X4/60.0)*SIGN(1.0,X3))
      IF(XLON.LT.0.0)XLON=XLON+360.0
C
C     OFFSET BY 5 MINUTES TO GET SQUARE CENTERED ON GRID
C     GET INTEGER LAT AND LON DEGREES
C
      LAT=XLAT+.08333333
      LON=XLON+.08333333
C
C     GET LAT AND LON MINUTES TO PRECISION OF TEN
C     ( 12.5 = 10, 21.73 = 20, ETC.)
C
      LATMIN=(INT((XLAT-FLOAT(LAT))*60.0+.5)/10.0)*10
      LONMIN=(INT((XLON-FLOAT(LON))*60.0+.5)/10.0)*10
C
C     IREC=LAT*360+MOD(LON+180,360)+1 WAS NOT USED AS
C     ADDRESSING SCHEME TO LOAD THE LAND MASK INFORMATION
C     DUE TO A BAD CASE OF 'HURRY UP' WHEN LOADING. (JCD)
C
      IREC=LAT*360+MOD(LON,360)+181
C
C     IF IN SAME DEGREE SQUARE AS LAST ACCESS ...
C
      IF(IPREC.EQ.IREC)THEN
C
```

```
C       IF IN THE SAME 10 MINUTE SQUARE AS LAST ACCESS ...
C
        IF(IPLATM.EQ.LATMIN.AND.IPLONM.EQ.LONMIN)THEN
C
C         RETURN SAME X5 AS LAST ACCESS
C
          X5=PX5
          RETURN
        END IF
      ELSE
C
C       IF NOT IN SAME DEGREE SQUARE AS LAST ACCESS, READ NEW RECORD
C
        READ(28,REC=IREC)ARRAY
      END IF
C
C     READ LAND MASK BITS FROM ARRAY AND SET X5 FOR LAND OR WATER
C
      J=0
      DO 10 I=2,20,6
        IF(LATMIN/10.EQ.J)THEN
          INDEX=I+LONMIN/10
          IF(BITS(ARRAY(INDEX),1,1).EQ.1)THEN
            X5=1.0
          ELSE
            X5=0.0
          END IF
          GO TO 30
        END IF
        J=J+1
   10 CONTINUE
      IBIT=19
      DO 20 I=1,7,6
        IF(LATMIN/10.EQ.J)THEN
          INDEX=I+LONMIN/10
          IF(LONMIN/10.EQ.0.AND.J.EQ.4)THEN
            INDEX=26
            IBIT=1
          END IF
          IF(BITS(ARRAY(INDEX),IBIT,1).EQ.1)THEN
            X5=1.0
          ELSE
            X5=0.0
          END IF
          GO TO 30
        END IF
        J=J+1
   20 CONTINUE
C
C     SAVE LAST ACCESS DATA
C
   30 IPREC=IREC
      IPLATM=LATMIN
      IPLONM=LONMIN
```

```
PX5=X5
RETURN
END
```

```
      SUBROUTINE NODAYS (IY,IM,ID,NDYM,NDY,IERR)
C     COMPUTES NODAYS IM MONTH, NDYM, AND
C     COMPUTES NUMBER OF THE DAY IN A YEAR, NDY, FROM
C     YEAR-IY, MONTH-IM, AND DAY OF MONTH-ID
      IERR = 0
      IF( IM-12)1,1,2
    2 WRITE (6,700)
  700 FORMAT(21H MON GREATER THAN 12  )
      IERR=1
      RETURN
    1 IF(IM) 3,3,4
    3 WRITE (6,705)
  705 FORMAT (  18H MONTH IS TO SMALL )
      IERR =1
      RETURN
    4 IF(ID) 5,6,6
    5 WRITE (6,710)
  710 FORMAT(12H DAY IS NEG  )
      IERR =1
      RETURN
    6 IF(IY) 7,8,8
    7 WRITE (6,715)
  715 FORMAT (13H YEAR IS NEG )
      IERR =1
      RETURN
    8 M = IM
      NDM =30+M+M/8-2*((M+M/8)/2)-((8-M)/6-(7-M)/6)*(1+(IY-4*(IY/4)+4)
     1 /5 )
      IF( ID) 9,9,10
    9 NDY =0
      NDYM = NDM
      RETURN
   10 IF( ID - NDM)11,11,12
   12 WRITE (6,720)
  720 FORMAT ( 17H DAY IS TO LARGE  )
      IERR= 1
      RETURN
   11 NDYM = NDM
      N= IM-1
      ISUM = 0
      IF(N) 30,30,35
   35 DO 15 M =1,N
      NDM =30+M+M/8-2*((M+M/8)/2)-((8-M)/6-(7-M)/6)*(1+(IY-4*(IY/4)+4)
     1 /5 )
   15 ISUM = ISUM + NDM
   30 ISUM = ISUM + ID
      NDY = ISUM
      RETURN
      END
```

```
C*********************************************************************
C
C   SUBROUTINE RDIN
C   THIS SUBROUTINE READS INPUT INSTRUCTIONS AND SETS DEFAULTS FOR
C   MOODSED.
C
C*********************************************************************
C
      SUBROUTINE RDIN(ISOL)
C
       CHARACTER*4 A
       CHARACTER*72 OUTFILE,INFILE,IN
C
       COMMON /VALIM /TMIN,TMAX,SMIN,SMAX,TEMP,SAL,
     &                TLO1,THI1,SLO1,SHI1,ZMIN
       COMMON /FILES/ INFILE,OUTFILE
       COMMON /CHKS/ LCHK,IDCK,ITOC,ISOC,ISBC,ITBC,IVCK,ZIVC,IECK,IBCK,
     &               IMIS,DMIS,STOL,IDUP,IMIN,DIST
       COMMON /INVERT/ DTOL,TTOL,BTOL
       COMMON /ZLIM/ ZMAX,ZVAL
C
       DATA TMIN,TMAX,SMIN,SMAX,ZMAX /-2.5,30.,33.,38.,6500./
       DATA TLO1,THI1,SLO1,SHI1,ZMIN /-2.5,12.,34.,37.,6500./
       DATA DTOL,TTOL,BTOL,ZIVC,DMIS,STOL/0.00001,1.,1.0,0.0,20.,3.5/
       DATA IMIN,DIST/60,10.0/
       DATA IVCK,IBCK,IECK/0,0,0/
       DATA LCHK,IDCK,ITOC,ISOC,ISBC,ITBC,IMIS,IDUP /0,0,0,0,0,0,0,0/
       DATA INFILE,OUTFILE /'[BILL.MOODS]MOODS','MOODS'/
C*********************************************************************
C
C   OPEN SCRATCH FILE TO ALLOW LIST DIRECTED READ
      CLOSE(UNIT=23)
      OPEN(UNIT=23,STATUS='SCRATCH',ERR=9096)
C
C   READ INPUT INTO INTERNAL FILE
 10    READ(5,5000,ERR=9000,END=999)IN
 5000  FORMAT(A)
C   WRITE INSTRUCTION TO UNIT 23
      WRITE(23,5000)IN(6:72)
      BACKSPACE 23
C
C*********************************************************************
C
C   TRANSLATE INSTRUCTION
      A=IN(1:4)
C
       IF(A(1:1).EQ.'H'.OR.A(1:1).EQ.'h')THEN
        ISOL=1
        RETURN
       END IF
C
       IF(A(1:2).EQ.'C '.OR.A(1:2).EQ.'c ')GO TO 10
C
       IF(A.EQ.'IFIL'.OR.A.EQ.'ifil')THEN
```

```
          INFILE=IN(6:72)
          GO TO 10
         END IF
C
         IF(A.EQ.'OFIL'.OR.A.EQ.'ofil')THEN
          OUTFILE=IN(6:72)
          GO TO 10
         END IF
C
         IF(A.EQ.'TMIN'.OR.A.EQ.'tmin')THEN
          READ(23,*,ERR=9097)TMIN
          GO TO 10
         END IF
C
         IF(A.EQ.'TMAX'.OR.A.EQ.'tmax')THEN
          READ(23,*,ERR=9097)TMAX
          GO TO 10
         END IF
C
         IF(A.EQ.'SMIN'.OR.A.EQ.'smin')THEN
          READ(23,*,ERR=9097)SMIN
          GO TO 10
         END IF
C
         IF(A.EQ.'SMAX'.OR.A.EQ.'smax')THEN
          READ(23,*,ERR=9097)SMAX
          GO TO 10
         END IF
C
C
         IF(A.EQ.'TLO1'.OR.A.EQ.'tlo1')THEN
          READ(23,*,ERR=9097)TLO1
          GO TO 10
         END IF
C
         IF(A.EQ.'THI1'.OR.A.EQ.'thi1')THEN
          READ(23,*,ERR=9097)THI1
          GO TO 10
         END IF
C
         IF(A.EQ.'SLO1'.OR.A.EQ.'slo1')THEN
          READ(23,*,ERR=9097)SLO1
          GO TO 10
         END IF
C
         IF(A.EQ.'SHI1'.OR.A.EQ.'shi1')THEN
          READ(23,*,ERR=9097)SHI1
          GO TO 10
         END IF
C
         IF(A.EQ.'ZMIN'.OR.A.EQ.'zmin')THEN
          READ(23,*,ERR=9097)ZMIN
          GO TO 10
         END IF
```

```
C
C
      IF(A.EQ.'ZMAX'.OR.A.EQ.'zmax')THEN
       READ(23,*,ERR=9097)ZMAX
       GO TO 10
      END IF
C
      IF(A.EQ.'LCHK'.OR.A.EQ.'lchk')THEN
       READ(23,*,ERR=9097)LCHK
       GO TO 10
      END IF
C
      IF(A.EQ.'IECK'.OR.A.EQ.'ieck')THEN
       READ(23,*,ERR=9097)IECK
       GO TO 10
      END IF
C
      IF(A.EQ.'IDCK'.OR.A.EQ.'idck')THEN
       READ(23,*,ERR=9097)IDCK
       GO TO 10
      END IF
C
      IF(A.EQ.'ITOC'.OR.A.EQ.'itoc')THEN
       READ(23,*,ERR=9097)ITOC
       GO TO 10
      END IF
C
      IF(A.EQ.'ISOC'.OR.A.EQ.'isoc')THEN
       READ(23,*,ERR=9097)ISOC
       GO TO 10
      END IF
C
      IF(A.EQ.'ISBC'.OR.A.EQ.'isbc')THEN
       READ(23,*,ERR=9097)ISBC
       GO TO 10
      END IF
C
      IF(A.EQ.'ITBC'.OR.A.EQ.'itbc')THEN
       READ(23,*,ERR=9097)ITBC
       GO TO 10
      END IF
C
      IF(A.EQ.'DTOL'.OR.A.EQ.'dtol')THEN
       READ(23,*,ERR=9097)DTOL
       GO TO 10
      END IF
C
      IF(A.EQ.'TTOL'.OR.A.EQ.'ttol')THEN
       READ(23,*,ERR=9097)TTOL
       GO TO 10
      END IF
C
      IF(A.EQ.'BTOL'.OR.A.EQ.'btol')THEN
       READ(23,*,ERR=9097)BTOL
```

```
        GO TO 10
       END IF
C
       IF(A.EQ.'IBCK'.OR.A.EQ.'ibck')THEN
        READ(23,*,ERR=9097)IBCK
        GO TO 10
       END IF
C
       IF(A.EQ.'IVCK'.OR.A.EQ.'ivck')THEN
        READ(23,*,ERR=9097)IVCK
        GO TO 10
       END IF
C
       IF(A.EQ.'ZIVC'.OR.A.EQ.'zivc')THEN
        READ(23,*,ERR=9097)ZIVC
        GO TO 10
       END IF
C
       IF(A.EQ.'IMIS'.OR.A.EQ.'imis')THEN
        READ(23,*,ERR=9097)IMIS
        GO TO 10
       END IF
C
       IF(A.EQ.'DMIS'.OR.A.EQ.'dmis')THEN
        READ(23,*,ERR=9097)DMIS
        GO TO 10
       END IF
C
       IF(A.EQ.'STOL'.OR.A.EQ.'stol')THEN
        READ(23,*,ERR=9097)STOL
        GO TO 10
       END IF
C
       IF(A.EQ.'IDUP'.OR.A.EQ.'idup')THEN
        READ(23,*,ERR=9097)IDUP
        GO TO 10
       END IF
C
       IF(A.EQ.'IMIN'.OR.A.EQ.'imin')THEN
        READ(23,*,ERR=9097)IMIN
        GO TO 10
       END IF
C
       IF(A.EQ.'DIST'.OR.A.EQ.'dist')THEN
        READ(23,*,ERR=9097)DIST
        GO TO 10
       END IF
C
C
C*************************************************************************
C
 9000  WRITE(6,*)'ERROR IN READING INSTRUCTION, TRY AGAIN'
       GO TO 10
 9096  WRITE(6,*)'ERROR IN OPENING UNIT 23, DO NOT USE UNIT 23!'
```

```
      STOP
 9097 WRITE(6,*)'ERROR WHILE READING INSTRUCTION ',A,' TRY AGAIN'
      GO TO 10
  999 RETURN
      END
```

```
      SUBROUTINE RDMOOD(D,T,S,NLEV,MAXLEV,IEOF)
C
C*********************************************************************
C   THIS PROGRAM READS THE MOODS DATA WHICH HAS BEEN EXTRACTED
C   FROM THE MOODS DATA BASE
C
C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
C!!!!!!!CHANGE THIS ROUTINE IF DIFFERENT INPUT FORMATS ARE REQUIRED!!!
C!!!!!!!CORRESPONDING CHANGES MUST THEN BE MADE TO THE WRITE ROUTINE!!
C!!!!!!!WEMOOD.!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
C*********************************************************************
C
      CHARACTER*13 GL1,GL2
      CHARACTER*131 INPUTS,INPUT
      CHARACTER*10 IP
C
      COMMON /RDCOM/ F(10),ID(10),TOP,BOT,ISEQ
      COMMON /RDCOMA/ INPUTS,GL1,GL2,IP
C
      DIMENSION D(1),T(1),S(1)
C
      DATA JPAS /0/
C
C*********************************************************************
C   READ MOODS DATA
 1    READ(27,5000,ERR=9000,END=900)INPUT
 5000 FORMAT(131A)
C
C   SKIP HEADER LABELS
      IF(INPUT(1:1).EQ.'L')THEN
       INPUTS=INPUT
C   INPUTS IS JUST A HEADER LINE, LAT,LONG, ETC.
C
C   GET PAGE NO.
      IP=INPUT(98:102)
      GO TO 1
      END IF
C
C   READ HEADER
C   LATITUDE: F(1), LONGITUDE: F(2), GL1: YEARMMDD HHMM,
C   GL2: SHIP KEY (IDENTIFYING LABEL), ID(4): DATA SOURCE CODE,
C   ID(5) AND ID(6): MOODS ERROR CODES, TOP: MIN DEPTH,
C   BOT: MAXIMIMUM DEPTH, NLEV: NO. OF DEPTH LEVELS,
C   ISEQ: CONSECUTIVE PROFILE SEQUENCE NO. BEGINNING WITH 1.
C
      READ(INPUT,5001,ERR=9000,END=900)F(1),F(2),GL1,GL2,
     & ID(2),ID(3),ID(4),ID(5),ID(6),TOP,
     & BOT,NLEV,ISEQ
 5001 FORMAT(1X,F6.2,F9.2,2X,A13,2X,A10,3I3,2(1X,02),1X,
     & 2F9.1,I6,2X,I6)
CW        WRITE(6,*)F(1),F(2),GL1,GL2,
CW     & ID(2),ID(3),ID(4),ID(5),ID(6),
CW     & TOP,BOT,NLEV
```

```
C
C   DO NOT ALLOW MORE THAN MAXLEV OF LEVELS
      IF(NLEV.GT.MAXLEV)THEN
       WRITE(6,*)'MAXIMUM NO. OF LEVELS EXCEEDED - ',MAXLEV,
     & 'PROFILE ',ISEQ
       WRITE(29,*)'MAXIMUM NO. OF LEVELS EXCEEDED - ',MAXLEV,
     & 'PROFILE ',ISEQ
       GO TO 9000
      END IF
C
C   READ DEPTHS   D(I)
       READ(27,5003,ERR=9000)(D(I),I=1,NLEV)
 5003  FORMAT(15F7.1)
CW       WRITE(6,*)(D(I),I=1,NLEV)
C
C   READ TEMPERATURES   T(I)
       READ(27,5002,ERR=9000)(T(I),I=1,NLEV)
 5002  FORMAT(1X,15F7.2)
CW       WRITE(6,*)(T(I),I=1,NLEV)
C
C   READ SALINITIES   S(I)
       READ(27,5002,ERR=9000)(S(I),I=1,NLEV)
CW       WRITE(6,*)(S(I),I=1,NLEV)
C
C   NORMAL RETURN
       IEOF=0
       RETURN
C   END OF FILE RETURN
 900   IEOF=1
       JPAS=JPAS+1
       WRITE(6,*)' END OF DATA REACHED, PASS',JPAS
       RETURN
C   ERROR RETURN
 9000  IEOF=2
       RETURN
       END
```

```
C*************************************************************************
C
C   SUBROUTINE SOLIC
C   THIS SUBROUTINE SOLICITATES INPUT INFORMATION WHEN REQUESTED BY
C   "HELP" INSTRUCTION.
C
C*************************************************************************
        SUBROUTINE SOLIC
C
C
        CHARACTER*72 OUTFILE,INFILE,QQ
        CHARACTER*4 YON
C
        COMMON /VALIM /TMIN,TMAX,SMIN,SMAX,TEMP,SAL,
     &                 TMIN1,TMAX1,SMIN1,SMAX1,ZMIN
        COMMON /FILES/ INFILE,OUTFILE
        COMMON /CHKS/ LCHK,IDCK,ITOC,ISOC,ISBC,ITBC,IVCK,ZIVC,IECK,IBCK,
     &                IMIS,DMIS,STOL,IDUP,IMIN,DIST
        COMMON /ZLIM/ ZMAX,ZVAL
        COMMON /INVERT/ DTOL,TTOL,BTOL,DELDEN,DELT
C
C*************************************************************************
C
        WRITE(6,*)' A BLANK RETURN PRESERVES DEFAULT VALUES'
        WRITE(6,*)' TYPE "CTRL Z" TO END SOLICITATION AND BEGIN PROGRAM'
     &  ,' EXECUTION.'
C   "CTRL Z" IS USED TO BRANCH FROM THE "END=" CONTAINED IN THE READ
C   STATEMENT, FOR THE  VAX COMPUTER.  "@END" IS USED FOR A UNIVAC COMPUTER.
C   THIS STATEMENT NEEDS TO BE TAILORIZED FOR THE PARTICULAR COMPUTER
C   BEING USED.
C
        WRITE(6,*)
        WRITE(6,*)'DEFAULT INPUT FILENAME IS: ',INFILE
        WRITE(6,*)
        WRITE(6,*)'INPUT FILENAME? '
        READ(5,555,END=999)QQ
 555    FORMAT(72A)
        IF(QQ.NE.' ')INFILE=QQ
        WRITE(6,*)'INPUT FILENAME IS: ',INFILE
C
        WRITE(6,*)
        WRITE(6,*)'DEFAULT OUTPUT FILENAME IS ',OUTFILE
        WRITE(6,*)'OUTPUT FILENAME?'
        READ(5,555,END=999)QQ
        IF(QQ.NE.' ')OUTFILE=QQ
        WRITE(6,*)'OUTPUT FILENAME IS ',OUTFILE
C
        WRITE(6,*)
        YON='YES'
        IF(ITBC.EQ.1)YON='NO'
        WRITE(6,*)'CHECK FOR BAD TEMPERATURES?  DEFAULT IS ',YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         ITBC=0
```

```
        YON='YES'
       END IF
       IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
        ITBC=1
        YON='NO'
       END IF
       WRITE(6,*)'CHECK FOR BAD TEMPERATURES? ',YON
C
      IF(YON.EQ.'YES')THEN
       WRITE(6,*)
       WRITE(6,*)'MINIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMIN
 90    WRITE(6,*)'NEW MINIMUM TEMPERATURE?'
       READ(5,555,END=999)QQ
        IF(QQ.NE.' ')THEN
        WRITE(29,555)QQ
        BACKSPACE 29
        READ(29,*,ERR=90)TMIN
        END IF
       WRITE(6,*)'MINIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMIN
C
       WRITE(6,*)
       WRITE(6,*)'MAXIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMAX
 91    WRITE(6,*)'NEW MAXIMUM TEMPERATURE?'
       READ(5,555,END=999)QQ
        IF(QQ.NE.' ')THEN
        WRITE(29,555)QQ
        BACKSPACE 29
        READ(29,*,ERR=91)TMAX
        END IF
       WRITE(6,*)'MAXIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMAX
      END IF
C
       WRITE(6,*)
       YON='YES'
       IF(ISBC.EQ.1)YON='NO'
       WRITE(6,*)'CHECK FOR BAD SALINITIES? DEFAULT IS ',YON
       READ(5,555,END=999)QQ
       IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
        ISBC=0
        YON='YES'
       END IF
       IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
        ISBC=1
        YON='NO'
       END IF
       WRITE(6,*)'CHECK FOR BAD SALINITIES? ',YON
C
      IF(YON.EQ.'YES')THEN
       WRITE(6,*)
       WRITE(6,*)'MINIMUM SALINITY ALLOWED IS ',SMIN
 92    WRITE(6,*)'NEW MINIMUM SALINITY?'
       READ(5,555,END=999)QQ
        IF(QQ.NE.' ')THEN
        WRITE(29,555)QQ
```

```
          BACKSPACE 29
          READ(29,*,ERR=92)SMIN
          END IF
         WRITE(6,*)'MINIMUM SALINITY ALLOWED IS ',SMIN
C
         WRITE(6,*)
         WRITE(6,*)'MAXIMUM SALINITY ALLOWED IS ',SMAX
 93      WRITE(6,*)'NEW MAXIMUM SALINITY?'
         READ(5,555,END=999)QQ
          IF(QQ.NE.' ')THEN
          WRITE(29,555)QQ
          BACKSPACE 29
          READ(29,*,ERR=93)SMAX
          END IF
         WRITE(6,*)'MAXIMUM SALINITY ALLOWED IS ',SMAX
        END IF
C
         WRITE(6,*)
         WRITE(6,*)'MAXIMUM DEPTH (M) ALLOWED IS ',ZMAX
 94      WRITE(6,*)'NEW MAXIMUM DEPTH?'
         READ(5,555,END=999)QQ
          IF(QQ.NE.' ')THEN
          WRITE(29,555)QQ
          BACKSPACE 29
          READ(29,*,ERR=94)ZMAX
          END IF
         WRITE(6,*)'MAXIMUM DEPTH (M) ALLOWED IS ',ZMAX
C
C
         WRITE(6,*)
         YON='YES'
         IF(IVCK.EQ.1)YON='NO'
         WRITE(6,*)'CHECK FOR TEMP/DENSITY INVERSIONS?  DEFAULT IS '
     &   ,YON
         READ(5,555,END=999)QQ
         IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
          IVCK=0
          YON='YES'
         END IF
         IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
          IVCK=1
          YON='NO'
         END IF
         WRITE(6,*)'CHECK FOR TEMP/DENSITY INVERSIONS? ',YON
C
        IF(YON.EQ.'YES')THEN
         WRITE(6,*)
         WRITE(6,*)'MAXIMUM TEMP. INVERSION (DEG C) ALLOWED IS ',TTOL
 95      WRITE(6,*)'NEW MAXIMUM TEMPERATURE INVERSION?'
         READ(5,555,END=999)QQ
          IF(QQ.NE.' ')THEN
          WRITE(29,555)QQ
          BACKSPACE 29
          READ(29,*,ERR=95)TTOL
```

```
          END IF
          WRITE(6,*)'MAXIMUM TEMP. INVERSION (DEG C) ALLOWED IS ',TTOL
C
          WRITE(6,*)
          WRITE(6,*)'MAXIMUM DENSITY INVERSION (GM/CM**3) ALLOWED IS ',
     &    DTOL
 96       WRITE(6,*)'NEW MAXIMUM DENSITY INVERSION?'
          READ(5,555,END=999)QQ
           IF(QQ.NE.' ')THEN
           WRITE(29,555)QQ
           BACKSPACE 29
           READ(29,*,ERR=96)DTOL
           END IF
          WRITE(6,*)'MAXIMUM DENSITY INVERSION (GM/CM**3) ALLOWED IS ',
     &    DTOL
C
          WRITE(6,*)
          WRITE(6,*)'CHECK FOR INVERSION BEYOND DEPTH ',ZIVC,' M'
 97       WRITE(6,*)'NEW DEPTH?'
          READ(5,555,END=999)QQ
           IF(QQ.NE.' ')THEN
           WRITE(29,555)QQ
           BACKSPACE 29
           READ(29,*,ERR=97)ZIVC
           END IF
          WRITE(6,*)'DEPTH BOUND FOR INVERSION CHECK ',ZIVC,' M'
         END IF
C
          WRITE(6,*)
          YON='YES'
          IF(IBCK.EQ.1)YON='NO'
          WRITE(6,*)'CHECK FOR DEPTHS GREATER THAN BOTTOM?  DEFAULT IS '
     &    ,YON
          READ(5,555,END=999)QQ
          IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
           IBCK=0
           YON='YES'
          END IF
          IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
           IBCK=1
           YON='NO'
          END IF
          WRITE(6,*)'CHECK FOR DEPTHS GREATER THAN BOTTOM? ',YON
C
         IF(YON.EQ.'YES')THEN
          WRITE(6,*)
          WRITE(6,*)'BOTTOM CHECKING - PERMISSIBLE DEPTH ERROR (%) ',BTOL
 98       WRITE(6,*)'NEW DEPTH ERROR (%)?'
          READ(5,555,END=999)QQ
           IF(QQ.NE.' ')THEN
           WRITE(29,555)QQ
           BACKSPACE 29
           READ(29,*,ERR=98)BTOL
           END IF
```

```
        WRITE(6,*)'MAXIMUM DEPTH ERROR (%) ALLOWED IS ',BTOL
       END IF
C
        WRITE(6,*)
        YON='YES'
        IF(LCHK.EQ.1)YON='NO'
        WRITE(6,*)'CHECK FOR DATA OVER LAND?  DEFAULT IS ',YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         LCHK=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         LCHK=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK FOR DATA OVER LAND? ',YON
C
        WRITE(6,*)
        YON='YES'
        IF(IDCK.EQ.1)YON='NO'
        WRITE(6,*)'CHECK FOR NEGATIVE, AND NON-MONOTONIC DEPTHS?'
        WRITE(6,*)'DEFAULT IS ',YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         IDCK=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         IDCK=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK FOR NEGATIVE AND NON-MONOTONIC DEPTHS? ',YON
C
        WRITE(6,*)
        YON='YES'
        IF(IECK.EQ.1)YON='NO'
        WRITE(6,*)'CHECK MOODS DUPLICATE PROFILE ERR CODE?  DEFAULT IS '
     &  ,YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         IECK=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         IECK=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK MOODS DUPLICATE PROFILE ERR CODE? ',YON
C
        WRITE(6,*)
        YON='YES'
        IF(ITOC.EQ.1)YON='NO'
        WRITE(6,*)'CHECK IF TEMPERATURES ARE ALL ZERO?  DEFAULT IS '
```

```
     &  ,YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         ITOC=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         ITOC=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK IF TEMPERATURES ARE ALL ZERO? ',YON
C
        WRITE(6,*)
        YON='YES'
        IF(ISOC.EQ.1)YON='NO'
        WRITE(6,*)'CHECK IF SALINITIES ARE ALL ZERO? DEFAULT IS '
     &  ,YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         ISOC=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         ISOC=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK IF SALINITIES ARE ALL ZERO? ',YON
C
        WRITE(6,*)
        YON='YES'
        IF(IMIS.EQ.1)YON='NO'
        WRITE(6,*)'CHECK FOR MISPLACED PROFILES? DEFAULT IS ',YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         IMIS=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         IMIS=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK FOR MISPLACED PROFILES? ',YON
C
       IF(YON.EQ.'YES')THEN
        WRITE(6,*)
        WRITE(6,*)'MAX STANDARD DEVIATION IS ',STOL
 99     WRITE(6,*)'NEW MAX STANDARD DEVIATION?'
        READ(5,555,END=999)QQ
         IF(QQ.NE.' ')THEN
         WRITE(29,555)QQ
         BACKSPACE 29
         READ(29,*,ERR=99)STOL
         END IF
        WRITE(6,*)'MAX STANDARD DEVIATION IS ',STOL
```

```
C
        WRITE(6,*)
        WRITE(6,*)'MAX DEPTH OF NEAR SURFACE TEMP IS ',DMIS,' M'
 100    WRITE(6,*)'NEW MAX DEPTH?'
        READ(5,555,END=999)QQ
         IF(QQ.NE.' ')THEN
         WRITE(29,555)QQ
         BACKSPACE 29
         READ(29,*,ERR=100)DMIS
         END IF
        WRITE(6,*)'MAX DEPTH OF NEAR SURFACE TEMP IS ',DMIS,' M'
       END IF
C
        WRITE(6,*)
        YON='YES'
        IF(IDUP.EQ.1)YON='NO'
        WRITE(6,*)'CHECK FOR DUPLICATE PROFILES? DEFAULT IS ',YON
        READ(5,555,END=999)QQ
        IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')THEN
         IDUP=0
         YON='YES'
        END IF
        IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')THEN
         IDUP=1
         YON='NO'
        END IF
        WRITE(6,*)'CHECK FOR DUPLICATE PROFILES? ',YON
C
      IF(YON.EQ.'YES')THEN
        WRITE(6,*)
        WRITE(6,*)'MAX TIME IN MINUTES BETWEEN PROFILES IS ',IMIN
 105    WRITE(6,*)'NEW MAX TIME?'
        READ(5,555,END=999)QQ
         IF(QQ.NE.' ')THEN
         WRITE(29,555)QQ
         BACKSPACE 29
         READ(29,*,ERR=105)IMIN
         END IF
        WRITE(6,*)'MAX TIME IN MINUTES BETWEEN PROFILES IS ',IMIN
C
        WRITE(6,*)
        WRITE(6,*)'MAX DISTANCE IN KM BETWEEN PROFILES IS ',DIST
 106    WRITE(6,*)'NEW MAX DISTANCE?'
        READ(5,555,END=999)QQ
         IF(QQ.NE.' ')THEN
         WRITE(29,555)QQ
         BACKSPACE 29
         READ(29,*,ERR=106)DIST
         END IF
        WRITE(6,*)'MAX DISTANCE IN KM BETWEEN PROFILES IS ',DIST
       END IF
C
        WRITE(6,*)
        YON='NO'
```

```
      WRITE(6,*)'CHANGE TEMP AND/OR SAL LIMITS AT DEPTH? ',
     & 'DEFAULT IS ',YON
C
      READ(5,555,END=999)QQ
      IF(QQ(1:1).EQ.'Y'.OR.QQ(1:1).EQ.'y')YON='YES'
      IF(QQ(1:1).EQ.'N'.OR.QQ(1:1).EQ.'n')YON='NO'
      WRITE(6,*)'CHANGE TEMP AND/OR SAL LIMITS AT DEPTH? ',YON
C
      IF(YON.EQ.'YES')THEN
      WRITE(6,*)
 110  WRITE(6,*)'NEW DEPTH FOR CHANGING TEMP/SAL LIMITS? ',
     &          'DEFAULT IS ',ZMIN
      READ(5,555,END=999)QQ
       IF(QQ.NE.' ')THEN
       WRITE(29,555)QQ
       BACKSPACE 29
       READ(29,*,ERR=110)ZMIN
       END IF
      IF(ZMIN.LE.0)GO TO 110
      WRITE(6,*)'NEW DEPTH FOR CHANGING TEMP/SAL LIMITS IS ',ZMIN
      WRITE(6,*)
      WRITE(6,*)'ENTER NEW TEMPERATURE/SALINITY LIMITS'
      WRITE(6,*)
      WRITE(6,*)'MINIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMIN1
 112  WRITE(6,*)'NEW MINIMUM TEMPERATURE?'
      READ(5,555,END=999)QQ
       IF(QQ.NE.' ')THEN
       WRITE(29,555)QQ
       BACKSPACE 29
       READ(29,*,ERR=112)TMIN1
       END IF
      WRITE(6,*)'MINIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMIN1
C
      WRITE(6,*)
      WRITE(6,*)'MAXIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMAX1
 114  WRITE(6,*)'NEW MAXIMUM TEMPERATURE?'
      READ(5,555,END=999)QQ
       IF(QQ.NE.' ')THEN
       WRITE(29,555)QQ
       BACKSPACE 29
       READ(29,*,ERR=114)TMAX1
       END IF
      WRITE(6,*)'MAXIMUM TEMPERATURE (DEG C) ALLOWED IS ',TMAX1
C
      WRITE(6,*)
      WRITE(6,*)'MINIMUM SALINITY ALLOWED IS ',SMIN1
 116  WRITE(6,*)'NEW MINIMUM SALINITY?'
      READ(5,555,END=999)QQ
       IF(QQ.NE.' ')THEN
       WRITE(29,555)QQ
       BACKSPACE 29
       READ(29,*,ERR=116)SMIN1
       END IF
      WRITE(6,*)'MINIMUM SALINITY ALLOWED IS ',SMIN1
```

```
C
        WRITE(6,*)
        WRITE(6,*)'MAXIMUM SALINITY ALLOWED IS ',SMAX1
 118    WRITE(6,*)'NEW MAXIMUM SALINITY?'
        READ(5,555,END=999)QQ
         IF(QQ.NE.' ')THEN
         WRITE(29,555)QQ
         BACKSPACE 29
         READ(29,*,ERR=118)SMAX1
         END IF
        WRITE(6,*)'MAXIMUM SALINITY ALLOWED IS ',SMAX1
       END IF
C
 999    RETURN
        END
```

```
      SUBROUTINE SORT(X,IY,NUM)
C   THIS ROUTINE SORTS  X AND IY BASED ON VARIABLE X
      DIMENSION X(1),IY(1)
C
C   IF NO DATA RETURN
      IF(NUM.EQ.0)RETURN
C
C   SORT VARIABLE X
      IF(NUM.LE.1)GO TO 999
      DO 150 I=1,NUM
      IFLAG=0
      DO 140 J=1,NUM-1
      IF(X(J).GT.X(J+1))THEN
       W=X(J+1)
       Z=IY(J+1)
       X(J+1)=X(J)
       X(J)=W
       IY(J+1)=IY(J)
       IY(J)=Z
       IFLAG=1
      END IF
  140 CONTINUE
      IF(IFLAG.EQ.0)GO TO 999
  150 CONTINUE
C
  999   RETURN
        END
```

```
      SUBROUTINE STATS(ISF,DAT,VMEAN,SDEV,NNUM)
C
C
      REAL*4 VMEAN,SDEV
      REAL*8 SUMSQ,SUM,VTERM1,VTERM2,VTERM3
      DATA SUMSQ,SUM,NUM/0.D0,0.D0,0/
C
      IF(ISF.EQ.1)GO TO 50
C
C      UPDATE STATS
       IF(DAT.LT.-998.)RETURN
       SUM=SUM+DAT
       SUMSQ=DAT*DAT+SUMSQ
       NUM=NUM+1
      RETURN
C
C   FINALIZE STATS
 50   CONTINUE
C   NUM MUST BE GREATER THAN 3
      IF(NUM.LE.3)THEN
       WRITE(6,*)'NOT ENOUGH DATA FOR MEANINGFUL STANDARD DEVIATION'
       WRITE(6,*)'NEED AT LEAST 4 GOOD PROFILES'
       WRITE(6,*)'TURN OFF MISPLACED PROFILE TEST'
       STOP
      END IF
      VMEAN=SUM/NUM
      SDEV=SUMSQ/NUM
            N=NUM
            VTERM1=-(N*1.0D0/(N*1.0-1.0))*1.0D0*VMEAN*VMEAN
            VTERM3=SUMSQ/(1.0D0*N-1.0D0)
            VTERM2=VTERM1+VTERM3
            SDEV=DSQRT(DABS(VTERM2))
      NNUM=NUM
      RETURN
      END
```

```
C*************************************************************************
C
C    SUBROUTINE VALUEC
C    THIS SUBROUTINE CHECKS FOR BAD TEMPERATURE AND SALINITY VALUES
C
C*************************************************************************
        SUBROUTINE VALUEC(D,T,S,NLEV,IVFLG)
C
        COMMON /VALIM/ TMIN1,TMAX1,SMIN1,SMAX1,TEMP,SAL,
     &                 TMIN2,TMAX2,SMIN2,SMAX2,ZMIN
        DIMENSION D(1),T(1),S(1)
C
C    SET MIN AND MAX
        TMIN=TMIN1
        TMAX=TMAX1
        SMIN=SMIN1
        SMAX=SMAX1
C    RESET FLAGS
        IVFLG=0
        IVFLGT=0
        IVFLGS=0
        IVFLGA=0
        IVFLGB=0
        ICHG=0
C    CHECK FOR ALL ZERO VALUES.
C    IF SALINITIES ARE ALL ZERO, IVFLG=1
C    IF TEMPERATURES ARE ALL ZERO, IVFLG=2
C    IF BOTH ARE ALL ZERO, IVFLG=3
        NT=0
        NS=0
        DO 50 I=1,NLEV
C
C    CHECK FOR ZERO VALUES
        IF(T(I).GT.-0.001.AND.T(I).LT.0.001)NT=NT+1
        IF(S(I).GT.-0.001.AND.S(I).LT.0.001)NS=NS+1
  50    CONTINUE
C
        IF(NS.EQ.NLEV.OR.NT.EQ.NLEV)THEN
         IF(NS.EQ.NLEV)THEN
          IVFLGS=1
C    CHANGE SALINITIES TO MISSING VALUES
          DO 60 I=1,NLEV
 60       S(I)=-999.0
         END IF
C
         IVFLG=IVFLGS
         IF(NT.EQ.NLEV)IVFLGT=2
         IF(IVFLGT.EQ.2)IVFLG=2
         IF(IVFLGS.EQ.1.AND.IVFLGT.EQ.2)IVFLG=3
        END IF
C
        IF(IVFLG.EQ.3)GO TO 105
C    CHECK FOR VALUES OUTSIDE OF BOUNDS
        DO 100 I=1,NLEV
```

```
C
C    CHANGE TEMP AND SAL LIMITS AT DEPTH
        IF(ICHG.EQ.0)THEN
         IF(D(I).GE.ZMIN)THEN
          TMIN=TMIN2
          TMAX=TMAX2
          SMIN=SMIN2
          SMAX=SMAX2
          ICHG=1
         END IF
        END IF
C
        IF(IVFLGT.EQ.2.OR.IVFLGA.EQ.1)GO TO 80
        IF(T(I).LT.-998.9.AND.T(I).GT.-999.1)GO TO 80
        IF(T(I).LT.TMIN.OR.T(I).GT.TMAX)THEN
        IVFLGA=1
        TEMP=T(I)
        END IF
C
 80     IF(IVFLGS.EQ.1.OR.IVFLGB.EQ.1)GO TO 100
        IF(S(I).LT.-998.9.AND.S(I).GT.-999.1)GO TO 100
        IF(S(I).LT.SMIN.OR.S(I).GT.SMAX)THEN
        IVFLGB=1
        SAL=S(I)
        END IF
C
 100    CONTINUE
C    IVFLG=4 FOR BAD TEMP, TEMP AND SAL ARE NON-ZERO
        IF(IVFLGA.EQ.1.AND.IVFLGS.EQ.0)IVFLG=4
C    IVFLG=5 FOR BAD SAL, TEMP AND SAL ARE NON-ZERO
        IF(IVFLGB.EQ.1.AND.IVFLGT.EQ.0)IVFLG=5
C    IVFLG=6 FOR BAD SAL AND BAD TEMP, TEMP AND SAL ARE NON-ZERO
        IF(IVFLGB.EQ.1.AND.IVFLGA.EQ.1)IVFLG=6
C    IVFLG=7 FOR BAD SAL, AND ALL ZERO TEMP
        IF(IVFLGB.EQ.1.AND.IVFLGT.EQ.2)IVFLG=7
C    IVFLG=8 FOR BAD TEMP, AND ALL ZERO SAL
        IF(IVFLGA.EQ.1.AND.IVFLGS.EQ.1)IVFLG=8
 105    RETURN
        END
```

```
      SUBROUTINE WEMOOD(D,T,S,NLEV,IEOF)
C
C*********************************************************************
C   THIS PROGRAM WRITES THE MOODS DATA WHICH HAS BEEN EDITED
C   AFTER EXTRACTON FROM THE MOODS DATA BASE
C
C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
C!!!THIS ROUTINE MUST BE CHANGED IF A DIFFERENT DATA FORMAT IS!!!!!!!!
C!!!REQUIRED.  SEE DOCUMENTATION CONTAINED IN RDMOOD!!!!!!!!!!!!!!!!!!
C!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
C*********************************************************************
C
      CHARACTER*13 GL1,GL2
      CHARACTER*131 INPUTS
      CHARACTER*10 IP
C
      COMMON /RDCOM/ F(10),ID(10),TOP,BOT,ISEQ
      COMMON /RDCOMA/ INPUTS,GL1,GL2,IP
C
      DIMENSION D(1),T(1),S(1)
C
      DATA LINE/71/,IPAGE/1/
C
C*********************************************************************
C   WRITE EDITED MOODS OUTPUT FILE
C
C   WRITE HEADER LABEL
      LINC=((NLEV-1)/15+1)*3+1
      IF(LINE+LINC.LE.70)GO TO 65
      LINE=2
      WRITE(INPUTS(98:102),501)IPAGE
  501 FORMAT(I5)
      WRITE(21,6440)INPUTS
 6440 FORMAT(131A)
      IPAGE=IPAGE+1
   65 CONTINUE
C
C   WRITE HEADER
      WRITE(21,5012,ERR=9004)F(1),F(2),GL1,GL2,
     & ID(2),ID(3),ID(4),ID(5),ID(6),TOP,
     & BOT,NLEV,ISEQ
 5012 FORMAT(1X,F6.2,F9.2,2X,A13,2X,A10,3I3,2(1X,02),1X,2F9.1,I6,
     & 2X,I6,2X,'----')
C
C   WRITE DEPTHS
      WRITE(21,5013,ERR=9004)(D(I),I=1,NLEV)
 5013 FORMAT(15F7.1)
C
C   WRITE TEMPERATURES
      WRITE(21,5014,ERR=9004)(T(I),I=1,NLEV)
 5014 FORMAT(1X,15F7.2)
C
C   WRITE SALINITIES
        WRITE(21,5014,ERR=9004)(S(I),I=1,NLEV)
```

```
C
      LINE=LINE+LINC
C
C   NORMAL RETURN
      IEOF=0
      RETURN
C   ERROR RETURN
 9004 IEOF=2
      RETURN
      END
```

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

AD-A167048

REPORT DOCUMENTATION PAGE

1a REPORT SECURITY CLASSIFICATION Unclassified	1b RESTRICTIVE MARKINGS None
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b DECLASSIFICATION/DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S) NORDA Report 136	5 MONITORING ORGANIZATION REPORT NUMBER(S) NORDA Report 136
6 NAME OF PERFORMING ORGANIZATION Naval Ocean Research and Development Activity	7a NAME OF MONITORING ORGANIZATION Naval Ocean Research and Development Activity
6c ADDRESS (City, State, and ZIP Code) Ocean Science Directorate NSTL, Mississippi 39529–5004	7b ADDRESS (City, State, and ZIP Code) Ocean Science Directorate NSTL, Mississippi 39529–5004
8a NAME OF FUNDING/SPONSORING ORGANIZATION Naval Ocean Research and Development Activity 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code) Ocean Science Directorate NSTL, Mississippi 39529–5004	10 SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO. 63704N; PROJECT NO.; TASK NO.; WORK UNIT NO.

11 TITLE (Include Security Classification)
A Data Base Editor for MOODS

12 PERSONAL AUTHOR(S)
William J. Teague, Robert L. Pickett, and Donald A. Burns

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Yr., Mo., Day)	15 PAGE COUNT
Final	From _____ To _____	February 1986	73

16 SUPPLEMENTARY NOTATION

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB GR	temperature, salinity, inversions

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

The Master Oceanographic Observations Data Set (MOODS) contains 3.5 million data observations. The data, which include temperature and salinity profiles, are edited during updates, but this editing has been very superficial and allows for erroneous values. This editor attempts to ferret out the bad data by checking for oceanographic observations that are over land, above the sea surface, below the sea bottom; that have nonmonotonic, duplicate, or negative depths; that contain impossible or all-zero temperatures or salinities; that produce temperature or density inversions; that are misplaced either by location or by season; or that are duplicates. For four MOODS test sets (two Atlantic and two Pacific), the total rejection rate ranged 17–39%. Of these rejections, 9–16% were already flagged during update editing, 1–8% were rejected because of inversions and wild values, and 7–17% were duplicate and misplaced profiles.

20 DISTRIBUTION AVAILABILITY OF ABSTRACT	21 ABSTRACT SECURITY CLASSIFICATION	
UNCLASSIFIED UNLIMITED ☐ SAME AS RPT ■ DTIC USERS ☐	Unclassified	
22a NAME OF RESPONSIBLE INDIVIDUAL William J. Teague	22b TELEPHONE NUMBER (Include Area Code) (601) 688-4734	22c OFFICE SYMBOL Code 331

DD FORM 1473, 83 APR EDITION OF 1 JAN 73 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

END

FILMED

6-86

DTIC