Department of Pesticide Regulation

Mary-Ann Warmerdam
Director

M E M O R A N D U M

Arnold Schwarzenegger
Governor

TO: Randy Segawa
Environmental Program Manager I
Environmental Monitoring Branch

FROM: Bruce Johnson, Ph.D. *Original signed by*
Research Scientist III
Environmental Monitoring Branch
916-324-4106

DATE: April 29, 2010

SUBJECT: DOCUMENTATION FOR WEATH6A METEOROLOGICAL DATA PROCESSING PROGRAM

Attached is the documentation for the WEATH6A meteorological data processing program. In brief, this program takes data-logger meteorological data and summarizes it, typically by hour, and provides a nearly ISCST3-ready file for modeling use. A second output file contains an expanded format which makes the manual determination of stability classes easier.

The most typical use of WEATH6A would be for site application studies when it is necessary to perform back-calculation procedures in order to estimate period flux density. This program has been compiled in Lahey Fortran LF95 and is available on the network drive at ISCST3-CLASS\T\WEATH6A.EXE. This version supersedes earlier (WEATH5) versions.

If there are problems running this program, feel free to contact me.

Attachment

Documentation for WEATH6A.EXE
Bruce Johnson, Ph.D.
Environmental Monitoring
April 28, 2010

WEATH6A is a weather data processing utility for assistance in creating ISC compatible meteorological data files from data logger files.

WEATH6A requires as input (1) the data logger meteorological values in comma separated format (2) (normally) a cut file which specifies the breakpoints for each time period for which average wind speed, temperature and direction are required (3) user-entry of sunrise and sunset times, wind direction measurement threshold, mixing height and user input ISC start time desired when the program is run.

WEATH6A provides two output files: (1) the data summary for each requested time period which is in ISC compatible format, but requires the user to input the stability classification column (2) an output file which provides more detail about the user input and the calculations for each period, and the classification of each period into day and night to help facilitate stability classification. These files are named by the user.

Input file details.

Meteorological data input file. The data logger input file is assumed to be in the following form (data from June 23 at 2:20 and 2:22 in the afternoon consisting of 2 dummy fields, and the first line of 74.53 *Fahrenheit* temperature, wind direction *from* 216 degrees, and wind speed of 12.72 *miles per hour*

6,23,14,20,1,74.53,1,216.4,12.72
6,23,14,22,1,73.56,1,221,11.12

mm,dd,hh,mi,int,at,agt,dir,spd where commas separate values and

mm=month	dd=day	hh=hour
mi=minute	int=dummyinterval	at=ambient temp (F)
agt=dummyholder	dir=FROM wind dir	spd=wind spd (mph)

Dummy fields not used, but must be present.

Here are two example lines: first line is June 23, 1400 hours 20 minutes, 74.53 F, wind from 216.4 degrees (from southwest) at 12.72 miles per hour.

6,23,14,20,1,74.53,1,216.4,12.72
6,23,14,22,1,73.56,1,221,11.12

Data lines must be sorted with oldest data first.

Here is a longer extract from the sample input data file:

```
6,23,14,20,1,74.53,1,216.40,12.72
6,23,14,22,1,73.56,1,221.00,11.12
6,23,14,24,1,73.47,1,232.50,9.01
6,23,14,26,1,73.20,1,234.30,10.75
6,23,14,28,1,72.73,1,244.40,10.42
6,23,14,30,1,72.59,1,233.20,10.30
6,23,14,32,1,73.06,1,250.80,10.47
6,23,14,34,1,72.88,1,251.10,10.22
6,23,14,36,1,72.64,1,246.30,11.11
6,23,14,38,1,72.19,1,244.80,10.83
6,23,14,40,1,72.30,1,235.00,10.36
6,23,14,42,1,72.79,1,240.30,10.82
6,23,14,44,1,72.81,1,237.70,10.43
6,23,14,46,1,72.90,1,249.90,10.80
6,23,14,48,1,72.68,1,247.10,10.75
6,23,14,50,1,72.70,1,247.60,10.98
```

Typically these data logger files are prepared in Excel. It may be necessary to convert degrees centigrade from the data logger to Fahrenheit and it may be necessary to convert meters per second from the data logger to miles per hour and rearrange the data into the order listed above, as well as adding the two dummy fields. The WEATH program was originally written utilizing English units because a long time ago, just after dinosaurs disappeared from the earth, most measurements that EM received were in miles per hour and Fahrenheit and the WEATH program has not been modified to be more flexible in this regard.

Cutpoint input file. The cutpoint file tells WEATH6A where to start the analysis and for what periods of time to summarize each of the meteorological parameters. The user must enter manually the start point for the first period for which the user desires to have an ISC met file created. WEATH6A will request this start point at the beginning of the setup. Again, this cutpoint file is typically prepared in Excel and saved as a CSV type file. Typically, also, the cutpoint file is designed to give hourly summaries during the monitoring event.

Here is an excerpt from a cutpoint file. This consists of month, day, hour, minute. Usually, minutes will be zero. This starts at June 24, hour 9. I usually start my cutpoint file with the first hour when monitoring started.

```
6, 24,  9,  0
6, 24, 10,  0
6, 24, 11,  0
6, 24, 12,  0
6, 24, 13,  0
```

6, 24, 14, 0
6, 24, 15, 0
6, 24, 16, 0
6, 24, 17, 0
6, 24, 18, 0
6, 24, 19, 0
6, 24, 20, 0
6, 24, 21, 0
6, 24, 22, 0
6, 24, 23, 0
6, 24, 24, 0
6, 25, 1, 0
6, 25, 2, 0
6, 25, 3, 0
6, 25, 4, 0
6, 25, 5, 0

Running WEATH6A and user-entry of sunrise/sunset times, threshold for wind direction measurements, cutfile. After a few introductory screens which describe input data variables and forms, the first entry is for the name of the comma-delimited meteorological data file to be processed. In this example, it is called 'sealmet.csv'. The format and variables for this file were described above. The second file, 'weath6a.out', is for documentation purposes. It is not the ISC meteorological output file that eventually will be made. Instead it is for purposes of documenting what the user entered into WEATH6A and gives more information about each period and night/day determinations which will be used later.

```
                    Press <RETURN> key to continue
 START BY ENTERING CSV INPUT WEATHER FILE...
--ENTER NAME OF FILE TO OPEN sealmet.csv
 ENTER DOCUMENTATION OUTPUT FILENAME.
NOTE-THIS IS NOT ISCST3 MET DATA FILE...
--ENTER NAME OF FILE TO OPEN weath6a.out
 FOLLOWING FILE EXISTS: weath6a.out


--OVERWRITE IT ?<Y/N> n
 --ENTER NAME OF FILE TO OPEN weath6ax.out
 ENTER SUNRISE HOUR&MIN AND SUNSET HOUR&MIN AS FOLLOWS: <MILITARY TIME!>
RH,RM,SH,SM   <IF MSMNTS IN PAC DAYLIGHT TIME, THEN
 GIVE SUNRISE SUNSET HOURS IN PACIFIC DAYLIGHT TIME>
```

In the example snippet above, the user first entered weath6a.out. WEATH6A determined that a file by this name already existed and asked the user if the user wished to overwrite the existing file. The user said no and entered a slightly different filename, 'weath6ax.out'. This shows what happens when there is an existing file that is in danger of being overwritten by WEATH6A.

The user must enter the sunrise and sunset times, using military time and in the same time system as the meteorological data. Sunrise and sunset times are available from many online sources. You will need the latitude and longitude. Just make sure that all of the time units are consistent (Pac Daylight Time vs Standard Time). If the meteorological data is in pacific daylight time, then so must the entered sunrise/sunset times. If the meteorological time is in standard time, then so must the sunrise/sunset times. The format is RH,RM,SH,SM which are sunrise hour, sunrise minute, sunset hour, sunset minute, all separated by commas. Here is a screen shot of entering this information:

```
                    Press <RETURN> key to continue
 START BY ENTERING CSV INPUT WEATHER FILE...
--ENTER NAME OF FILE TO OPEN sealmet.csv
 ENTER DOCUMENTATION OUTPUT FILENAME.
NOTE-THIS IS NOT ISCST3 MET DATA FILE...
--ENTER NAME OF FILE TO OPEN junk.out
 ENTER SUNRISE HOUR&MIN AND SUNSET HOUR&MIN AS FOLLOWS: (MILITARY TIME!)
RH,RM,SH,SM   (IF MSMNTS IN PAC DAYLIGHT TIME, THEN
 GIVE SUNRISE,SUNSET HOURS,IN PACIFIC DAYLIGHT TIME)  5,43,20,7
```

The last four numbers on the lower right were entered and then <RETURN> is pressed. These numbers represent sunrise at 5:43 and sunset at 20:07 in military time.

```
 ENTER SUNRISE HOUR&MIN AND SUNSET HOUR&MIN AS FOLLOWS: (MILITARY TIME!)
RH,RM,SH,SM   (IF MSMNTS IN PAC DAYLIGHT TIME, THEN
 GIVE SUNRISE,SUNSET HOURS,IN PACIFIC DAYLIGHT TIME)  5,43,20,7
 ENTER HEADER FOR OUTPUT Seal Beach Methyl Bromide Study

ENTER WIND SPEED THRESHOLD FOR DIRECTION IN MPH 2.2_
```

Next the program asks for a header (title) which should be some descriptive text (Seal Beach methyl bromide study, eg). And next you must enter a threshold speed for the wind direction. This is the speed below which the wind directions measurements are not reliable. In example above, it was 2.2 miles per hour.

Next the program will ask if the cutpoints are in a file. The answer will normally be Y for yes. And then you must enter the filename containing the cutpoints. This file must be a comma separated file, time sorted and with a format described above. In the above

```
 GIVE SUNRISE,SUNSET HOURS,IN PACIFIC DAYLIGHT TIME)  5,43,20,7
 ENTER HEADER FOR OUTPUT Seal Beach Methyl Bromide Study

ENTER WIND SPEED THRESHOLD FOR DIRECTION IN MPH 2.2

TODAY'S DATE: MAR 26 2010
        TIME: 11:29
   WEATH6A: LAST MODIFIED 12/17/2009. SCALER WIND SPEED,
      INCLUDES ALL WIND MSMNTS AND AVERAGE DIRECTION BASED
      ONLY ON MSMNTS WHERE SPEED> 2.2000MPH  (0.9834M/S).
-----------------------------------------------------------
--CUTPOINTS IN FILE ?(Y/N) y
 --ENTER NAME OF FILE TO OPEN cuthour.in_
```

example, the file was called *cuthour.in*. It doesn't matter if the file is upper or lower case letters.

Next the user enters a name for the soon-to-be-created ISC meteorological data file. In the example the name 'sbisc.met' is used. This is one of two output files that WEATH6A creates (the other one above was weath6a.out). Also, the program checks to see if this file

```
      INCLUDES ALL WIND MSMNTS AND AVERAGE DIRECTION BASED
      ONLY ON MSMNTS WHERE SPEED> 2.2000MPH  <0.9834M/S>.
-----------------------------------------------------------------
--CUTPOINTS IN FILE ?<Y/N> y
 --ENTER NAME OF FILE TO OPEN cuthour.in
 ENTER FILENAME TO CREATE ISCST3 MET DATA FILE
enter filename sbisc.met
```

already exists and if so, it will ask you if you wish to overwrite and gives you an opportunity to change the name if necessary.

```
--CUTPOINTS IN FILE ?<Y/N> y
 --ENTER NAME OF FILE TO OPEN cuthour.in
 ENTER FILENAME TO CREATE ISCST3 MET DATA FILE
enter filename sbisc.met

ENTER STARTING VALUES FOR YEAR,MONTH,DAY,HOUR
FOR FIRST RECORD OF ISCST3 MET CONTROL FILE
THE FORMAT IS YY,MM,DD,HH <FOR EXAMPLE, 90,3,25,15> 99,6,24,9_
```

Next you must enter the starting times for when you want WEATH6A to start the analysis. Note that since the input met file does NOT contain year, you are specifying the year by whatever year you enter at this point in the program. In this case, the year is 99, month is June (6), day is June 24, and the hour is 9 (9AM) (see screen shot below).

```
 ENTER FILENAME TO CREATE ISCST3 MET DATA FILE
enter filename sbisc.met

ENTER STARTING VALUES FOR YEAR,MONTH,DAY,HOUR
FOR FIRST RECORD OF ISCST3 MET CONTROL FILE
THE FORMAT IS YY,MM,DD,HH <FOR EXAMPLE, 90,3,25,15> 99,6,24,9

NOW ENTER MIXING HEIGHT IN METERS 300.
```

Next the mixing height (meters) is entered. Typically numbers of 300-350m are entered here. The WEATH6A program fixes the rural and urban mixing heights to whatever is entered here. In practice, for site specific application studies, the mixing height does not have any effect unless it is set very low.

```
NOW ENTER MIXING HEIGHT IN METERS 300.


CAREFULLY EXAMINE THE FOLLOWING DATA FROM
sealmet.csv
TO MAKE SURE THAT VARIABLES ARE READING CORRECTLY:
FIRST LINE: 6,23,14,20,1,74.53,1,216.40,12.72
PARSED AS FOLLOWS:
MM DD HH MI AMBTMP DIRECT SPEED
 6.23.14.20.  74.5 216.4  12.7

--OK TO PROCEED ?(Y/N)
```

Next WEATH6A takes the first record from the input CSV met file (in the example "sealmet.csv", interprets it and prints it out on the screen for you to look at and verify that the file is being read correctly and that the variables are in the correct format and order. And WEATH6A asks if it is OK to proceed.

Finally, type in 'y' to proceed with the analysis and 'y' to end it.

Output Files Details

There are two output files: one which contains the nearly-ready ISC compatible meteorological data, summarized from the input meteorological file over the periods specified by the cutpoint file and a second which documents various values used as input, provides more information about each summary and gives the night/day determinations.

Nearly-ready ISC compatible meteorological data file. Here are a few lines from the example run above (this was called SBISC.MET).

sbisc.met - Notepad

File Edit Format View Help

```
   99999       99    99999       99
99 624 9   68.6439    0.7367 292.7 X  300.0  300.0
99 62410   36.0762    3.5957 292.6 X  300.0  300.0
99 62411   49.7803    4.1236 292.9 X  300.0  300.0
99 62412   55.6602    4.5357 293.0 X  300.0  300.0
99 62413   48.5289    4.6531 294.4 X  300.0  300.0
99 62414   58.2812    5.4138 295.4 X  300.0  300.0
99 62415   60.9318    5.1967 296.2 X  300.0  300.0
99 62416   64.1871    6.1775 294.8 X  300.0  300.0
99 62417   63.9935    4.5952 294.5 X  300.0  300.0
99 62418  118.1664    3.5326 296.0 X  300.0  300.0
99 62419  132.4641    3.7558 293.4 X  300.0  300.0
99 62420  131.1457    2.8593 290.6 X  300.0  300.0
99 62421  137.8309    2.2669 289.6 X  300.0  300.0
99 62422  143.4305    2.3856 289.2 X  300.0  300.0
99 62423  146.0880    2.4971 289.0 X  300.0  300.0
99 62424  145.9636    1.0649 288.7 X  300.0  300.0
99 625 1  339.6841    0.7852 288.3 X  300.0  300.0
99 625 2   14.3680    0.7489 288.9 X  300.0  300.0
99 625 3    2.0674    0.9789 289.0 X  300.0  300.0
```

Using NOTEPAD to look at the file, the first line contains an arbitrary surface and upperair met station code (99999) and the year (99) for each. These MUST agree with the corresponding line in the ISCST3 input control file in the ME section.

The second and subsequent lines in the above named sbcisc.met give the meteorological data. For comparing to monitoring studies, these records will have to be broken up into periods corresponding to the measurement period. For example, suppose the first measurement period started on June 24 at 9AM and lasted 5 hours. Then the five records from 99 624 9 to 9972413 would be copied and pasted into a separate file along with the first record (99999 etc, these station ID numbers and year must be in any ISC met file). I might call the file PER01.MET, for example, to indicate that it was the meteorology for the first monitoring period. Note that the record '99 624 0 68.6439…' represents the summary from 9AM to 10AM and so would be the appropriate record to start covering the monitoring period with.

WEATH6A produces output file giving time interval, the average 'to' wind direction in degrees (mod(180+FROM,360)), scalar average wind speed (meters/second=.447*milesperhour), average ambient air temperature in degrees Kelvin (K=273+C), average wind direction TO (mod(180+FROM,360)), scaler average wind speed (meters/second=.447*milesperhour), a column of X, and the urban and rural mixing heights in meters. It is relatively easy to check the scalar average wind direction and temperature using Excel on the raw data. Use the 'average' worksheet function. Checking the direction is more difficult. See Appendix 2.

The column of Xs are for the user to enter the stability class (1-6) based on night/day, wind speed and sun angle.

Documentation file. The documentation file is meant to help document how the analysis was done and what the user-input values were.

The excerpt below shows the date and time that the file was created, the user-inputted sunrise and sunset times and threshold value. The time period used to generate each record is also shown. For example, the first line is from 6/24 9AM to 6/24 10AM. This file can be helpful for determining stability because it contains the night/day determinations.

weath6a.out - Notepad

File Edit Format View Help

```
WEATHER INPUT FILE: sealmet.csv
class example
USER DEFINED VALUES: SUNRISE@ 5:43, SUNSET@20: 7
     NOTE THAT DAY STARTS 1 HR AFTER SUNRISE AND
     ENDS 1 HR BEFORE SUNSET FOR STABILITY CALCS
TODAY'S DATE: MAR 26 2010
        TIME: 10:06
   WEATH6A: LAST MODIFIED 12/17/2009. SCALER WIND SPEED,
      INCLUDES ALL WIND MSMNTS AND AVERAGE DIRECTION BASED
      ONLY ON MSMNTS WHERE SPEED> 2.2000MPH  (0.9834M/S).
-------------------------------------------------------------
NAME OF ISCST3 MET DATA FILE: sbcisc.met
-------------------------------------------------------------

                              SCA AVG    180+AVG  SCA AVG    NUMBER OF
.......TIME PERIOD........    TEMPERA K  DIRECTI  SPD M/S    RECORDS    N/D
6/24  9: 0 TO  6/24 10: 0     292.698    68.644     0.737     ( 30)     DAY
6/24 10: 0 TO  6/24 11: 0     292.607    36.076     3.596     ( 30)     DAY
6/24 11: 0 TO  6/24 12: 0     292.893    49.780     4.124     ( 30)     DAY
6/24 12: 0 TO  6/24 13: 0     293.001    55.660     4.536     ( 30)     DAY
6/24 13: 0 TO  6/24 14: 0     294.423    48.529     4.653     ( 30)     DAY
6/24 14: 0 TO  6/24 15: 0     295.390    58.281     5.414     ( 30)     DAY
6/24 15: 0 TO  6/24 16: 0     296.229    60.932     5.197     ( 30)     DAY
6/24 16: 0 TO  6/24 17: 0     294.826    64.187     6.178     ( 30)     DAY
6/24 17: 0 TO  6/24 18: 0     294.535    63.993     4.595     ( 30)     DAY
6/24 18: 0 TO  6/24 19: 0     295.987   118.166     3.533     ( 30)     DAY
6/24 19: 0 TO  6/24 20: 0     293.402   132.464     3.756     ( 30)     NIGHT
6/24 20: 0 TO  6/24 21: 0     290.648   131.146     2.859     ( 30)     NIGHT
6/24 21: 0 TO  6/24 22: 0     289.615   137.831     2.267     ( 30)     NIGHT
6/24 22: 0 TO  6/24 23: 0     289.174   143.431     2.386     ( 30)     NIGHT
6/24 23: 0 TO  6/24 24: 0     288.967   146.088     2.497     ( 30)     NIGHT
6/24 24: 0 TO  6/25  1: 0     288.701   145.964     1.065     ( 30)     NIGHT
6/25  1: 0 TO  6/25  2: 0     288.292   339.684     0.785     ( 30)     NIGHT
6/25  2: 0 TO  6/25  3: 0     288.879    14.368     0.749     ( 30)     NIGHT
6/25  3: 0 TO  6/25  4: 0     288.965     2.067     0.979     ( 30)     NIGHT
6/25  4: 0 TO  6/25  5: 0     288.803   346.987     1.253     ( 30)     NIGHT
6/25  5: 0 TO  6/25  6: 0     288.634    10.897     1.612     ( 30)     NIGHT
6/25  6: 0 TO  6/25  7: 0     288.684     6.764     1.460     ( 30)     NIGHT
6/25  7: 0 TO  6/25  8: 0     288.989    21.844     1.059     ( 30)     DAY
6/25  8: 0 TO  6/25  9: 0     289.421     5.326     1.731     ( 30)     DAY
6/25  9: 0 TO  6/25 10: 0     290.263    49.704     1.519     ( 30)     DAY
```

The determination of stability class is a separate topic and won't be discussed here.

Appendix 1.

The latitude and longitude for Seal Beach is

33° 44' 29" N / 118° 6' 14" W

Seal Beach, CA

County: Orange County
State: California
Country: United States
Latitude: 33.7413889
Longitude: -118.1038889

The United States Navy Astronomical Applications Department also lets you enter a city without knowing the latitude and longitude in order to calculate the sunrise and sunset times.

http://aa.usno.navy.mil/

Go to

Table of Sunrise/Sunset, Moonrise/Moonset, or Twilight Times for an Entire Year

And fill in the fields.

The table below gives sunrise sunset times for Seal Beach, 1999, in Pacific Standard Time. Note that the sunrise time for June 24 is 0443 and the sunrise time used in the WEATH6A calculations was 0543. This is because the data logger times were all in Pacific Daylight Time and so the sunrise/sunset times were put into Pacific Daylight time in order to be consistent with the underlying data that was collected.

Sun or Moon Rise/Set Table for One Year - Mozilla Firefox

File Edit View History Bookmarks Tools Help

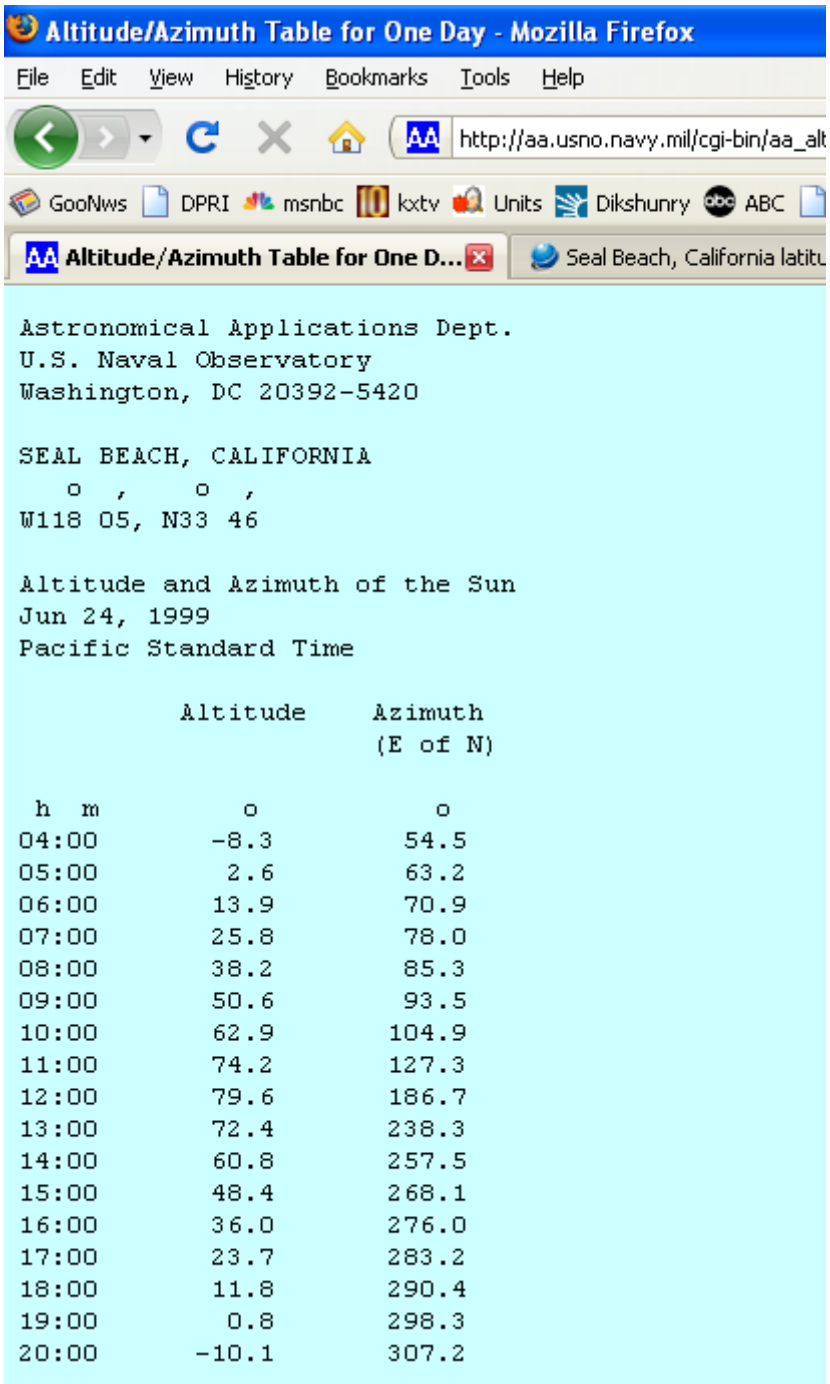
http://aa.usno.navy.mil/cgi-bin/aa_rstablew.pl

GooNws DPRI msnbc kxtv Units Dikshunry ABC TUM sbc PCprogress

Sun or Moon Rise/Set Table for O... | Seal Beach, California latitude/longitude

	Jan.		Feb.		Mar.		Apr.		May		June	
Day	Rise	Set	Rise	Set	Rise	Set	Rise	Set	Rise	Set	Rise	Set
	h m	h m	h m	h m	h m	h m	h m	h m	h m	h m	h m	h m
01	0657	1655	0649	1723	0621	1749	0540	1813	0504	1835	0443	1858
02	0657	1655	0648	1724	0620	1750	0539	1814	0503	1836	0443	1858
03	0658	1656	0648	1725	0619	1750	0538	1814	0502	1837	0442	1859
04	0658	1657	0647	1726	0617	1751	0536	1815	0501	1838	0442	1859
05	0658	1658	0646	1727	0616	1752	0535	1816	0500	1839	0442	1900
06	0658	1659	0645	1728	0615	1753	0534	1816	0459	1839	0442	1900
07	0658	1659	0644	1729	0614	1754	0532	1817	0458	1840	0442	1901
08	0658	1700	0644	1730	0612	1754	0531	1818	0457	1841	0441	1901
09	0658	1701	0643	1731	0611	1755	0530	1819	0456	1842	0441	1902
10	0658	1702	0642	1732	0610	1756	0528	1819	0455	1842	0441	1902
11	0658	1703	0641	1733	0608	1757	0527	1820	0455	1843	0441	1903
12	0658	1704	0640	1734	0607	1758	0526	1821	0454	1844	0441	1903
13	0657	1705	0639	1735	0606	1758	0525	1822	0453	1845	0441	1904
14	0657	1706	0638	1735	0604	1759	0523	1823	0452	1845	0441	1904
15	0657	1707	0637	1736	0603	1800	0522	1823	0452	1846	0441	1904
16	0657	1708	0636	1737	0602	1801	0521	1824	0451	1847	0441	1905
17	0657	1708	0635	1738	0600	1801	0520	1825	0450	1848	0441	1905
18	0656	1709	0634	1739	0559	1802	0518	1826	0450	1848	0442	1905
19	0656	1710	0633	1740	0558	1803	0517	1826	0449	1849	0442	1906
20	0656	1711	0632	1741	0556	1804	0516	1827	0448	1850	0442	1906
21	0655	1712	0631	1742	0555	1805	0515	1828	0448	1851	0442	1906
22	0655	1713	0629	1743	0554	1805	0514	1829	0447	1851	0442	1906
23	0654	1714	0628	1744	0552	1806	0512	1829	0447	1852	0443	1906
24	0654	1715	0627	1744	0551	1807	0511	1830	0446	1853	0443	1907
25	0653	1716	0626	1745	0550	1808	0510	1831	0446	1853	0443	1907
26	0653	1717	0625	1746	0548	1808	0509	1832	0445	1854	0443	1907

To get the sun angle during a day, go to Altitude and Azimuth of the Sun or Moon During One Day on the Navy website. The altitude is used in stability determinations.



Astronomical Applications Dept.
U.S. Naval Observatory
Washington, DC 20392-5420

SEAL BEACH, CALIFORNIA
o , o ,
W118 05, N33 46

Altitude and Azimuth of the Sun
Jun 24, 1999
Pacific Standard Time

h m	Altitude o	Azimuth (E of N) o
04:00	-8.3	54.5
05:00	2.6	63.2
06:00	13.9	70.9
07:00	25.8	78.0
08:00	38.2	85.3
09:00	50.6	93.5
10:00	62.9	104.9
11:00	74.2	127.3
12:00	79.6	186.7
13:00	72.4	238.3
14:00	60.8	257.5
15:00	48.4	268.1
16:00	36.0	276.0
17:00	23.7	283.2
18:00	11.8	290.4
19:00	0.8	298.3
20:00	-10.1	307.2

Appendix 2.

Calculating the average 'to' wind direction from the 'from' directions and wind speed. Here are the formulas used to calculate the various quantities.

Microsoft Excel - winddir-sample-calc.xls

A1 wind dir (after adding 14 deg for decl)

	A	B	C	D	E	F	G	H	I
1	wind dir (after adding 14 deg	speed mph	angle radians	speed m/s	speed indicator	sin(angle)	indic*speed(m/s)*sin	cos(angle)	indic*speed(m/s)*cos
2					2.2				
3	48.29	1.05	=A3*3.14159/180	=0.447*B3	=IF(B3<2.2,0,1)	=SIN(C3)	=D3*E3*F3	=COS(C3)	=E3*D3*H3
4	34.85	1.34	=A4*3.14159/180	=0.447*B4	=IF(B4<2.2,0,1)	=SIN(C4)	=D4*E4*F4	=COS(C4)	=E4*D4*H4
5	3	2.34	=A5*3.14159/180	=0.447*B5	=IF(B5<2.2,0,1)	=SIN(C5)	=D5*E5*F5	=COS(C5)	=E5*D5*H5
6	8.3	1.8	=A6*3.14159/180	=0.447*B6	=IF(B6<2.2,0,1)	=SIN(C6)	=D6*E6*F6	=COS(C6)	=E6*D6*H6
7	29.96	0.17	=A7*3.14159/180	=0.447*B7	=IF(B7<2.2,0,1)	=SIN(C7)	=D7*E7*F7	=COS(C7)	=E7*D7*H7
8	129.5	0.17	=A8*3.14159/180	=0.447*B8	=IF(B8<2.2,0,1)	=SIN(C8)	=D8*E8*F8	=COS(C8)	=E8*D8*H8

These are the formulas at the bottom of the worksheet for this one hour period.

27	=0.447*B27	=IF(B27<2.2,0,1)	=SIN(C27)	=D27*E27*F27	=COS(C27)	=E27*D27*H27
28	=0.447*B28	=IF(B28<2.2,0,1)	=SIN(C28)	=D28*E28*F28	=COS(C28)	=E28*D28*H28
29	=0.447*B29	=IF(B29<2.2,0,1)	=SIN(C29)	=D29*E29*F29	=COS(C29)	=E29*D29*H29
30	=0.447*B30	=IF(B30<2.2,0,1)	=SIN(C30)	=D30*E30*F30	=COS(C30)	=E30*D30*H30
31	=0.447*B31	=IF(B31<2.2,0,1)	=SIN(C31)	=D31*E31*F31	=COS(C31)	=E31*D31*H31
32	=0.447*B32	=IF(B32<2.2,0,1)	=SIN(C32)	=D32*E32*F32	=COS(C32)	=E32*D32*H32
33	=AVERAGE(D2:D32)					
34						
35		sum(i*s*sin)		=SUM(G2:G32)		=SUM(I2:I32)
36			atan	=ATAN2(I35,G35)		sum(I*s*cos)
37			to angle	=180*IF(G36<0,G36+2*3.14159,G36)/3.14159		
38			from angle	=MOD(G37-180,360)		
39						

In words, the wind direction angle corresponds to the sum of the speed weighted sines of angles divided by the sum of the speed weighted cosines of angles for those records where the speed was greater than the threshold (in this case 2.2 miles per hour).

$$from\,angle = \tan^{-1}\left[\frac{\sum_i I*\sin(\theta_i)*s_i}{\sum_i I*\cos(\theta_i)*s_i}\right]$$

where I is the indicator variable (=0, if speed < 2.2 mph, 1 otherwise), θ_i is the ith measured wind angle (converted to radians) and s_i is the ith measured wind speed in mph. The other worksheet functions convert from degrees to radians and back and insure that the angle is positive between 0 and 360, and reverse the direction in order to produce the 'to' direction.

wind dir (after adding 14 deg for decl)	speed mph	angle radians	speed m/s	speed indicator	sin(angle)	indic*speed (m/s)*sin	cos(angle)	indic*speed(m/s)*cos
				2.2				
48.29	1.05	0.84	0.47	0.00	0.75	0.00	0.67	0.00
34.85	1.34	0.61	0.60	0.00	0.57	0.00	0.82	0.00
3.00	2.34	0.05	1.05	1.00	0.05	0.05	1.00	1.04
8.30	1.80	0.14	0.80	0.00	0.14	0.00	0.99	0.00
29.96	0.17	0.52	0.08	0.00	0.50	0.00	0.87	0.00
129.50	0.17	2.26	0.08	0.00	0.77	0.00	-0.64	0.00
114.80	1.32	2.00	0.59	0.00	0.91	0.00	-0.42	0.00
149.90	0.85	2.62	0.38	0.00	0.50	0.00	-0.87	0.00
153.80	2.31	2.68	1.03	1.00	0.44	0.46	-0.90	-0.93
157.80	1.01	2.75	0.45	0.00	0.38	0.00	-0.93	0.00
346.80	1.19	6.05	0.53	0.00	-0.23	0.00	0.97	0.00
307.30	0.25	5.36	0.11	0.00	-0.80	0.00	0.61	0.00
275.60	1.87	4.81	0.84	0.00	-1.00	0.00	0.10	0.00
274.90	1.30	4.80	0.58	0.00	-1.00	0.00	0.09	0.00
217.70	1.05	3.80	0.47	0.00	-0.61	0.00	-0.79	0.00
295.70	0.79	5.16	0.35	0.00	-0.90	0.00	0.43	0.00
61.93	0.99	1.08	0.44	0.00	0.88	0.00	0.47	0.00
75.35	1.48	1.32	0.66	0.00	0.97	0.00	0.25	0.00
65.70	1.90	1.15	0.85	0.00	0.91	0.00	0.41	0.00
88.80	1.94	1.55	0.87	0.00	1.00	0.00	0.02	0.00
63.78	1.91	1.11	0.85	0.00	0.90	0.00	0.44	0.00
52.27	2.79	0.91	1.25	1.00	0.79	0.99	0.61	0.76
41.28	2.01	0.72	0.90	0.00	0.66	0.00	0.75	0.00
290.80	1.15	5.08	0.51	0.00	-0.93	0.00	0.36	0.00
229.70	3.75	4.01	1.68	1.00	-0.76	-1.28	-0.65	-1.08
246.90	4.70	4.31	2.10	1.00	-0.92	-1.93	-0.39	-0.82
253.30	3.18	4.42	1.42	1.00	-0.96	-1.36	-0.29	-0.41
261.00	2.27	4.56	1.01	1.00	-0.99	-1.00	-0.16	-0.16
205.70	0.79	3.59	0.35	0.00	-0.43	0.00	-0.90	0.00
220.20	1.77	3.84	0.79	0.00	-0.65	0.00	-0.76	0.00
			0.74					
				sum(i*s*sin)		-4.08		-1.59
					atan	-1.94		sum(I*s*cos)
					to angle	248.64		
					from angle	68.64		

Appendix 3. Program listing for WEATH6A.FOR

Main program

WEATH6A

Subroutine list

```
ASK
bjPACK
DLINE
FLUSHC
M1A
M1A_95
M1B_95
M2
M2A
MAK3DO
MAK3IN
NDET
OF
OPENE
OPENN
PAUS
STAB
T
TCALC
TW4
```

```
C     Last change:  BJ   17 Dec 2009    1:56 pm
      PROGRAM WEATH6A
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C 091210 WORKED ON THIS PACKAGE
C UPDATED TO FORTRAN95
C WHICH MOSTLY CONSISTED OF
C CHANGING READ/WRITES TO UNIT 0 TO UNITS 5,6
C CHANGING STRINGS TO 200 UNIFORMLY (SUCH AS IN UTILITIES)
C CHANGING SOMEWHAT THE WAY THAT ASK IS CALLED (MESS='  ' BEFORE ASK
CALL, THEN
C               MESS IS ARGUMENT TO ASK
C ADDING INTENT STATEMENTS
C RENAMING SOME VARIABLES (UNIT,MIN,LEN) WHICH CONFLICTED WITH
INTRINSICS IN FORTRAN 95
C
C I SYSTEMATICALLY CHANGED FIRST THE SET OF UTILITIES, WRITING CALLING
ROUTINES
C AND TESTING EACH ONE, THEN I EDITED THE WEATH6 SUBROUTINES.  HOWEVER,
I DID NOT
C WRITE CALLING ROUTINES TO TEST THESE, SO KEEP THAT IN MIND.
C THEN I WENT THROGH WEATH6 AND EDITED AND COMPILED, THEN WILL TEST
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCCCCC
c$DEBUG
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C 3/2/01 TESTED FOR 0 OR NEGATIVE VALUE IN MAGNITUDE VECTOR BEFORE
TAKING SQRT
C
C 6/24/97 SEVERAL CHANGES NEED TO BE MADE TO THE PROGRAM
C     1. REMOVE PRINTOUT OF STANDARD DEVIATION AND STAB CLASS
C     2. LEAVE DAY/NIGHT DESIGNATOR
C     3. CREATE NEW FILE WITH STANDARD ISCST3 MET OUTPUT FORMAT
C     4. USE SCALER WIND AVERAGE, INSTEAD OF VECTOR
C     5. USE ALL WIND SPEEDS TO DETERMINE SCALER AVERAGE
C     6. FOR WIND SPEEDS BELOW DIRECTION THRESHOLD (WHICH WILL BE
C        INPUT BY THE USER), SET WIND SPEED EQUAL TO 0 FOR PURPOSES
C        OF DETERMING THE AVERAGE DIRECTION
C
C 3/11/94 VARIABLES,UNITS,FORMAT: AS A HELPFUL NOTE:
C VARIABLE IN INPUT SHOULD BE AS FOLLOWS:
C   MONTH,DAY,HOUR,MINUTE,X,AMB TEMP (F),X,'FROM' WIND DIRECTION,WIND
SPEED (MPH)
C   THE X STANDS FOR A PLACE HOLDER:3,10,12,30,,70,,10,5
C   WOULD MEAN MARCH 10 AT 12:30, 70 DEGREES FAHRENHEIT, WIND FROM 10
DEGREES
C   (JUST EAST OF NORTH), AND SPEED OF 5 MPH
C
C
C 6/2/93 EXAMINED THE QUESTION OF WHETHER OR NOT TO INCLUDE 0 WIND
C SPEED MEASUREMENTS IN CALCULATIONS.  FOR WIND ANGLE, INCLUDING 0
C WIND SPEED DOES NOT AFFECT RESULTANT ANGLE.  HOWEVER, FOR MAGNITUDE
C OF THE VECTOR, IT DOES AFFECT THE MAGNITUDE OF THE VECTOR.  IT'S THE
C DIFFERENCE BETWEEN TAKING AN AVERAGE OF THE POSITIVE WIND SPEED
RECORDS
C VS TAKING AN AVERAGE OF POSITIVE AND 0 WIND SPEED RECORDS.  ACCORDING
```

```
C TO THE CIMIS MANUAL, THEY SEEM TO CALCULATE AN AVERAGE BASED ON
C ALL OF THE RECORDS, NOT JUST POSITIVE ONES.  SO THAT IS WHAT I'LL DO
C THE AFFECT WILL BE TO LOWER THE MAGNITUDE WHEN THERE ARE RECORDS WITH
C 0 WIND SPEEDS PRESENT.
C
C 5/26/93 MODIFIED TO CORRECT ERROR WHEN UPPER BOUND OF TIME INTERVAL
C COINCIDED WITH DATA RECORD, THAT RECORD WOULD GET COUNTED IN EVEN
THOUGH
C IT WASN'T ACTUALLY IN TIME INTERVAL (IE NEEDED LOWER LE DATA LT
UPPER, NOT
C LOWER LE DATA LE UPPER)  THIS WOULD HAVE AFFECTED ONLY CASES WHERE
DATA VALUE
C AND UPPER BOUND THE SAME.  IN THOSE AFFECTED CASES, THE GREATEST
AFFECT
C WOULD HAVE BEEN WITH FEWER SAMPLES PER TIME INTERVAL, MAY WISH TO
RECHECK
C THE UCD STUFF AND LET EVERYONE ELSE KNOW - I REVISED BRITZ WEATHER
PROGRAM
C IN FACT, IT WAS IN REVIEW OF ANALYSIS OF THE BRITZ MET DATA ANALYSIS
THAT
C I DISCOVERED THIS PROBLEM, MUCH OF THE OTHER SIEMER DATA DO NOT HAVE
C COINCIDENT DATA AND INTERVAL VALUES
C
C 5/15/93 MODIFIED TO LOOK FOR 0 WIND SPEED, WHICH
C CAUSES THAT RECORD TO GET SKIPPPED
C
C 2/17/93 MODIFIED TO CHECK FOR 0,0 IN ATAN2 ARGUMENT
C WHICH OCCURRED WHEN ALL OF THE WIND SPEEDS WERE 0.
C PRINTOUT NOW GIVES ***** WHERE THIS HAPPENS
C
C TO ALLOW EASIER SUMMARY OF MET DATA FROM
C MEBR STUDIES
C
C 2/1/93 WEATH4: I MODIFIED WEATH3 TO CALCULATE THE
C MEAN VECTOR AND STANDARD DEVIATION ACCORDING TO EQUATIONS
C SUPPLIED BY DWR FROM CIMIS CUSTODIANS.
C
C THESE EQUATIONS ARE MEAN DIRECTION REQUIRES FOR
C EACH MEASUREMENT OF WIND SPEED AND DIRECTION, MULTIPLYING
C THE COS (SIN) BY THE SPEED AND ARCTANING THAT TO GET THE
C DIRECTION.
C
C THE MEAN SPEED IS CALCULATED AS JUST THE MEAN OF THE SPEEDS
C
C THE STANDARD DEVIATION OF THE ANGLE IS CALCULATED WITH
C SD(THETA)=81*(1-(UBAR/SBAR))**0.5 WHERE UBAR IS THE
C MAGNITUDE OF THE RESULTANT VECTOR WHEN ALL THE SPEED VECTORS
C ARE ADDED TOGETHER AND SBAR IS THE MEAN OF THE SPEEDS
C I KEEP THIS EXPLANATION IN CIMIS VOLUME
C
C
C
C WEATH3 12/14/92, MODIFIED WEATH1 ADDED STANDARD DEVIATION
C CALCULATION OF ANGLES, AND ESTIMATE THE STABILITY CLASS
C BASED ON ZANETTI PGS 148-149, TABLES 7-2 AND 7-3 FOR
C DETERMINING STABILITY CLASSES BASED ON STANDARD DEVIATION
C OF HORIZONTAL WIND ANGLE AND NIGHT VS DAYTIME PLUS
```

```
C WINDSPEED
C ZANETTI, PAOLO. 1990. AIR POLLUTION MODELING. VAN NOSTRAND
C AND REINHOLD, NEW YORK.
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL SRH,SRM,SSH,SSM
      REAL DUMM01  !added 3/5/01 to trap for negative sqrt

C SUNRISE HOUR, MINUTE, SUNSET HOUR, MINUTE

      REAL ARRANG(1000),SD,TMPANG,STANA

C ARRANG STORES WIND DIRECTIONS TO DETERMINE SD, STANDARD DEVIATION
C OF WIND DIRECTION

      REAL VARS(9),ANGLE

C VARS CONTAINS THE VARIABLES TRANSLATED FROM THE DATA LINE

      REAL SMONTH,SDAY,SHOUR,SMIN,INTERV,STIM
      REAL EMONTH,EDAY,EHOUR,EMIN,ETIM
      REAL NMONTH,NDAY,NHOUR,NMIN,NTIM

C START MONTH, START DAY, ETC

      REAL AIRT,SUMAIR,WDEG,SUMSIN,SUMCOS,SPEED,SUMSPEED
      REAL TMPSPD, DIRTHR, DIRTMS !FOR WEATH5 MODS, TEMPORARY SPEED,
AND DIR THRESHOLD

C FOLLOWING ADDED IN PROGRAMMING WEATH4, MAGNITUDE OF
C RESULTANT VECTOR
      REAL MAGVEC

C AIR TEMPERATURE, SUMS, DIRECTION,

      REAL DIRSIN,DIRCOS
      REAL ATAN2,TCALC
      CHARACTER*200 F1,F2,F3
      LOGICAL OPENE,OPENN,T,FIRST,ASK,ENGAG,JUSTPR,ENDOF
      LOGICAL C0
      LOGICAL CUTFIL,TW4

C ENGAG,JUSTPR,ENDOF VARIOUS FLAGS TO CONTROL PROCESSING

      CHARACTER*80 TITLE
      CHARACTER*200 LINE,MESS
      character*8 note

      LOGICAL NDET,NIGHT

C NIGHT IS TRUE IF NIGHTTIME MEASUREMENTS, NDET DETERMINES IF
C SOMETHING IS AT NIGHT OR NOT DEPENDING ON USER SET SUNRISE,
C SUNSET VALUES

      CHARACTER*5 DN
```

```
C8/12/97 STUFF ADDED FOR ISCST3 MET DATA FILE CREATION
      LOGICAL OF  !LOGICAL FUNCTION TO OPEN NEW FILE
      REAL MIX
      CHARACTER*200 FMAK3 !THIS FILE IS ISCST3 MET FILE CREATED

      FIRST=.TRUE.
      JUSTPR=.FALSE.
      C0=.FALSE.
      ENDOF=.FALSE.
        DUMLEN=200
C following for debuggin purposes
C     OPEN(FILE='test.OUT',STATUS='NEW',UNIT=8)  !DEBUGGING FILE
      CALL M1A_95
c     WRITE(6,50)
c50   FORMAT(1X,'***********************************')
      CALL PAUS
        CALL M1B_95
        CALL PAUS

C GET INPUT FILE NAME FIRST

1     WRITE(6,100)
100   FORMAT(1X,'START BY ENTERING CSV INPUT WEATHER FILE...  ')
C     READ(5,110)F1
C110  FORMAT(A200)
        CALL FLUSHC(F1,200) !OPENE ZEROS THIS OUT ANYWAY, SO LET OPENE
HANDLE INPUT
      IF(.NOT.OPENE(1,F1,DUMLEN))THEN
       WRITE(6,120)F1(1:58)
120    FORMAT(1X,'UNABLE TO OPEN ',A58)
       GOTO1
      ENDIF

C ESTABLISH OUTPUT FILENAME

2     WRITE(6,150)
150   FORMAT(1X,'ENTER DOCUMENTATION OUTPUT FILENAME. ',/1X,'NOTE-',
     1            'THIS IS NOT ISCST3 MET DATA FILE... ')
C     READ(5,110)F2  !LET OPENN HANDLE THE INPUT
        CALL FLUSHC(F2,200)
      IF(.NOT.OPENN(2,F2,DUMLEN))THEN
       WRITE(6,170)F2(1:58)
170    FORMAT(1X,'UNABLE TO OPEN FOR OUTPUT:  ',A58)
       GOTO2
      ENDIF

C WRITE OUT PROGRAM ID AND NAME OF INPUT FILE
C     CALL M1A(2,DIRTHR) !8/5/97 NOPE, WAIT TIL GET DIRTHR
C     CALL M1A(0,dirthr)
      WRITE(2,7241)F1(1:58)
7241  FORMAT(1X,'WEATHER INPUT FILE: ',A58)

C GET SUNRISE AND SUNSET HOURS AND MINUTES FOR USE DETERMINING
C STABILITY CLASS

179   WRITE(6,180)
180   FORMAT(1X,'ENTER SUNRISE HOUR&MIN AND SUNSET HOUR&'
```

```
     1,'MIN AS FOLLOWS: (MILITARY TIME!)'
     2,/1X,'RH,RM,SH,SM','   (IF MSMNTS IN PAC DAYLIGHT TIME, THEN',
     1/1x,' GIVE SUNRISE,SUNSET HOURS,'
     1,'IN PACIFIC DAYLIGHT TIME)  ',1X\)
      READ(5,*,ERR=179)SRH,SRM,SSH,SSM
c185  FORMAT(4F10.0)
C THE ASTERISK FORMAT NOW APPEARS NECESSARY TO REPLACE THE SHORT FIELD
TERMINATION
C THAT USED TO WORK (I.E. 4F10.0 NO LONGER WORKS)

C RUN PRELIMINARY CHECK ON THESE VALUES JUST TO MAKE SURE
C THAT THEY'RE NOT OUTLANDISH, USE CHECKING ROUTINES INSIDE
C OF NDET, EVEN THOUGH DON'T REALLY CARE AT THIS POINT WHAT
C NDET EVALUATION IS.  NDET WILL STOP PROCESSING IF THERE'S
C A PROBLEM

      NIGHT=NDET(1.,1.,SRH,SRM,SSH,SSM)

C GET TITLE FOR OUTPUT FILE

      WRITE(6,200)
200   FORMAT(1X,'ENTER HEADER FOR OUTPUT ')
      READ(5,210)TITLE
210   FORMAT(A80)
      WRITE(2,220)TITLE
220   FORMAT(1X,A80)

C WRITE OUT SUNRISE, SUNSET HOURS

      WRITE(2,233)INT(SRH),INT(SRM),INT(SSH),INT(SSM)
233   FORMAT(1X,'USER DEFINED VALUES: SUNRISE@',I2,':',I2,
     1', SUNSET@',I2,':',I2,
     2/1X,'     NOTE THAT DAY STARTS 1 HR AFTER SUNRISE AND',
     3/1X,'     ENDS 1 HR BEFORE SUNSET FOR STABILITY CALCS')
C     CALL M2

C GET DIRECTION THRESHOLD SPEED, IDEA FOR WEATH5 8/5/97 MODS THAT
C FOR SPEEDS BELOW THRESHOLD, A ZERO WIND SPEED WILL BE ASSIGNED BEFORE
C DETERMINING DIRECTIONAL SIN AND COSIN (IE THESE DIRECTIONS WILL NOT
BE
C USED IN DETERMINING AVERAGE DIRECTION)
      WRITE(6,277)
277   FORMAT(1X,'ENTER WIND SPEED THRESHOLD FOR DIRECTION IN MPH ')
      READ(5,278)DIRTHR
278   FORMAT(F10.0)

C WRITE OUT PROGRAM ID AND NAME OF INPUT FILE
      CALL M1A(2,DIRTHR)
      CALL M1A(6,dirthr)
C     CALL M2

C GET START AND END FOR FIRST TIME PERIOD
      TCX=0
        CALL FLUSHC(MESS,200)
        MESS(1:18)='CUTPOINTS IN FILE '
7477  CUTFIL=ASK(MESS,18)
      IF (CUTFIL)THEN
```

```
       IF(.NOT.OPENE(7,F3,DUMLEN))THEN
        TCX=TCX+1
        IF(TCX.GT.3)THEN
          WRITE(6,7497)TCX
7497      FORMAT(1X,'SORRY ABORTING AFTER ',I3,' TRIES')
          STOP
         ENDIF
         WRITE(6,7571)F3(1:58)
7571     FORMAT(1X,'UNABLE TO OPEN ',A58)
           CALL FLUSHC(MESS,200)
           MESS(1:10)='TRY AGAIN '
         IF(ASK(MESS,10))THEN
          GOTO 7477
         ELSE
          STOP
         ENDIF
        ENDIF

      ENDIF
C
C FOLLOWING ADDED 8/12/97 FOR GETTING FILENAME FOR ISCST3 MET DATA FILE
      WRITE(6,4772)
4772  FORMAT(1X,'ENTER FILENAME TO CREATE ISCST3 MET DATA FILE ')
      IF(.NOT.OF(3,FMAK3))THEN
        WRITE(6,4773)
4773    FORMAT(1X,'PROGRAM IS TERMINATING, UNABLE OPEN ISCST3 MET ',
     1              'FILE')
        STOP
      ENDIF

C WRITE OUT NAME OF ISCST3 MET DATA FILE TO DOCUMENTATION FILE
      WRITE(2,6115)FMAK3(1:80)
6115  FORMAT(1X,'NAME OF ISCST3 MET DATA FILE: ',A80)

C WRITE OUT LINE OF DASHES TO SEPARATE EXPLANATORY TEXT
      CALL DLINE(2)

C WRITE OUT HEADERS FOR DOCUMENTATION FILE
      CALL M2

C GET STARTING DATE AND HOUR AND MIXING HEIGHT FOR ISCST3 MET DATA FILE
      CALL MAK3IN(YY,MM,DD,HH,MIX)
      WRITE(3,4873)'99999',YY,'99999',YY  !WRITE 1ST LINE OF ISCST3 MET
FILE
4873  FORMAT(T4,A5,T15,I2,T20,A5,T31,I2)  !END OF 8/12/97 ADDITIONS

      IF(.NOT.CUTFIL)THEN
3      WRITE(6,300)
300    FORMAT(1X,'ENTER START DATE/TIME FIRST PERIOD ',
     1/1X,'MM,DD,HH,MM')
       IF(.NOT.T(SMONTH,SDAY,SHOUR,SMIN,STIM))GOTO3
      ELSE
       IF(.NOT.TW4(SMONTH,SDAY,SHOUR,SMIN,STIM,7))STOP
      ENDIF

      IF(.NOT.CUTFIL)THEN
4      WRITE(6,310)
```

```
310    FORMAT(1X,'ENTER END DATE/TIME FIRST PERIOD ',
     1/1X,'MM,DD,HH,MM')
       IF(.NOT.T(EMONTH,EDAY,EHOUR,EMIN,ETIM))GOTO4
      ELSE
       IF(.NOT.TW4(EMONTH,EDAY,EHOUR,EMIN,ETIM,7))STOP
      ENDIF


C GOT START AND END DATE/TIME, NOW BEGIN READING

      COUNT=0
      DAYCNT=0

C DAYCNT COUNTS THE NUMBER OF DATA LINES WHICH ARE IN THE
C DAY TIME PORTION OF THE DAY (1 HOUR AFTER SUNRISE, 1 HOUR
C BEFORE SUNSET) IN ORDER TO DETERMINE IF THIS SET OF MEASUREMENTS
C IS MOSTLY DAY OR MOSTLY NIGHT

      SUMAIR=0.
      SUMSIN=0.
      SUMCOS=0.
      SUMSPEED=0.
        ENGAG=.FALSE.

10    CONTINUE

C FOLLOWING BLOCK OCCURS EVERYTIME AFTER FIRST SECTION DONE

      IF(ENGAG)THEN
       IF(.NOT.CUTFIL)THEN
5       WRITE(6,380)
380     FORMAT(1X,'ENTER END DATE/TIME (CNTRL Z TO END)',
     1/1X,'MM,DD,HH,MI')
        IF(.NOT.T(EMONTH,EDAY,EHOUR,EMIN,ETIM))GOTO5
       ELSE
        IF(.NOT.TW4(EMONTH,EDAY,EHOUR,EMIN,ETIM,7))STOP
       ENDIF
       ENGAG=.FALSE.
      ENDIF

C NOW BEGIN PROCESSING FILE, BUT DON'T DO FOLLOWING READ
C IF HAVE LINE LEFTOVER FROM HAVING CLOSED OUT LAST
C TIME BLOCK

      IF(.NOT.JUSTPR)THEN
       READ(1,400,END=1000,ERR=2000)LINE
400    FORMAT(A200)
C      write(0,6010)line
C6010  format(1x,'line= ',a60)
       CALL bjPACK(LINE,200)
      ENDIF

      READ(LINE,*,END=3000,ERR=4000)VARS
C420  FORMAT(9F10.0)

C FOLLOWING BLOCK RESETS STARTING TIME AT BEGINNING
C OF NEXT TIME SECTION, ONLY OCCURS AFTER TIME SECTION
```

```
C HAS JUST BEEN PRINTED OUT

      IF(JUSTPR)THEN
       SMONTH=VARS(1)
       SDAY=VARS(2)
       SHOUR=VARS(3)
       SMIN=VARS(4)
       JUSTPR=.FALSE.
      ENDIF

C CHECK FORMATTING TO BE SURE, IF THIS IS FIRST TIME

      IF(FIRST)THEN
       WRITE(6,487)F1
487    FORMAT(//1X,'CAREFULLY EXAMINE THE FOLLOWING DATA FROM ',
     1/1X,A40,/1X,'TO MAKE SURE THAT VARIABLES ARE READING CORRECTLY:')
       WRITE(6,501)LINE(1:60),(VARS(K),K=1,4),VARS(6),
     1    VARS(8),VARS(9)
501    FORMAT(1X,'FIRST LINE: ',A60,/1X,'PARSED AS FOLLOWS:'
     1    /1X,'MM DD HH MI AMBTMP DIRECT SPEED',
     2    /1X,4F3.0,3F6.1/)
         CALL FLUSHC(MESS,200)
         MESS(1:14)='OK TO PROCEED '
       IF(.NOT.ASK(MESS,14))STOP
       FIRST=.FALSE.
      ENDIF

C SET VARIABLES EQUAL TO APPROPRIATE VALUES

      NMONTH=VARS(1)
      NDAY=VARS(2)
      NHOUR=VARS(3)
      NMIN=VARS(4)
      AIRT=273.16+(5./9.)*(VARS(6)-32.)
      WDEG=2.*3.1415927*VARS(8)/360.
C     DIRSIN=SIN(WDEG)  !THIS WAS STATEMENT IN WEATH3
C     DIRCOS=COS(WDEG)  !THIS WAS STATEMENT IN WEATH3
      SPEED=0.447*VARS(9)
      DIRTMS=0.447*DIRTHR !CONVERT WIND SPEED DIR THRESHOLD FROM MPH TO
MS
C     if(speed.eq.0.)then
C       note='not used'
C     else
C       note='        '
C     endif
c     write(8,2993)vars(1),vars(2),vars(3),vars(8),vars(9),note
c2993 format(1x,5f8.2,1x,a8)

cccccccccccc6/2/93 i don't think this is necessary anymore, because
c if speed is 0, then the the angle shouldn't be added into the total
c and therefore wont affect the result - so take these lines out and
check
c to see if anything affected
c
C LATER 6/2/93...OK SOMETHING WAS AFFECTED, THE WIND SPEED, BUT AFTER
C REVIEWING CIMIS MATERIALS, IT APPEARS THAT YOU SHOULD INCLUDE 0 WIND
C SPEEDS IN AVERAGE SO WILL LEAVE THESE LINES COMMENTED OUT
```

```
C
C FOR BRITZ DATA, CHECK TO SEE IF SPEED IS 0, IF SO, THEN
C FORGET THIS LINE OF DATA AND GO TO NEXT LINE
c     IF(SPEED.EQ.0.)THEN
c      WRITE(6,2994)LINE(1:50)
c2994  FORMAT(1X,'0 WINDSPEED, SKIP: ',A50)
c         GOTO10
c     ENDIF

C NOW CALCULATE COS/SIN AND MULTIPLY BY SPEED
C BUT ONLY USE MEASUREMENTS WHERE WIND SPEED WAS GREATER THAN THE
C THRESHOLD 8/5/97 FOR WEATH5, SPEED HAS ALREADY BEEN CONVERTED TO MS

      IF(SPEED.LT.DIRTMS)THEN
        TMPSPD=0.
C       write (2,71182)speed,dirtMS
C71182        format(1x,'speed set to 0: speed, dirtMS ',2f10.5)
      ELSE
        TMPSPD=SPEED
      ENDIF

      DIRSIN=TMPSPD*SIN(WDEG)
      DIRCOS=TMPSPD*COS(WDEG)

c     write(8,57791)nhour,nmin,360.*wdeg/(2.*3.1415927),wdeg,
c     1                speed,tmpspd,dirtms,dirsin,dircos,sumsin,sumcos
c57791      format(1x,2f4.0,1x,9f7.3)

C CHECK TO SEE IF THIS IS IN THE INTERVAL, FIRST GET TIME FROM
C START OF YEAR IN MINUTES

      NTIM=TCALC(NMONTH,NDAY,NHOUR,NMIN)

C ARE WE BEFORE THE TIME PERIOD

      IF(NTIM.LT.STIM)GOTO10

C ARE WE IN THE TIME PERIOD, IF SO, PROCESS, ADD TO TOTALS

c     IF(NTIM.GE.STIM.AND.NTIM.LE.ETIM)THEN  this incorrect changed
5/26/93
      IF(NTIM.GE.STIM.AND.NTIM.Lt.ETIM)THEN

       SUMAIR=SUMAIR+AIRT
       SUMSIN=SUMSIN+DIRSIN
       SUMCOS=SUMCOS+DIRCOS
C      write(2,1377)line(1:15),sumsin,dirsin,sumcos,dircos,wdeg
C1377  format(1x,'line(15),sumsin,dirsin,sumcos,dircos',/1x,
C     1a15,4f10.5,/1x,'wdeg= ',f10.2)
       SUMSPEED=SUMSPEED+SPEED

C IF THIS RECORD IS IN THE DAYTIME, THEN INCREMENT DAYCNT

      IF(.NOT.(NDET(NHOUR,NMIN,SRH,SRM,SSH,SSM)))THEN
       DAYCNT=DAYCNT+1
      ENDIF
```

```
C AND INCREMENT COUNT, TO KEEP TRACK OF THE NUMBER OF RECORDS
C USED FOR THESE SUMS

       COUNT=COUNT+1

C.............FOR WEATH 4 WE WON'T USE THIS, BUT DO IT ANYWAY.....
C FOLLOWING LINE ADDED FOR WEATH3, WILL DO STANDARD DEVIATION
C CALCULATION IN DEGREES INSTEAD OF RADIANS
       ARRANG(COUNT)=VARS(8)
C............................................................


       GOTO10
      ENDIF

C ARE WE BEYOND THE TIME PERIOD, IF SO, HOLD THIS LINE, AND
C PROCESS TIME PERIOD, THEN USE THIS LINE AS FIRST LINE
C FOR NEXT BLOCK TO PROCESS

C OR ARE WE AT END OF FILE? (ENDOF=TRUE)

c504  IF((NTIM.GT.ETIM).OR.ENDOF)THEN this incorrect, changed5/26/93
504   IF((NTIM.Ge.ETIM).OR.ENDOF)THEN

C CHECK FOR 0 COUNT (COULD OCCUR IF CUT POINTS NOT RIGHT OR SOMETHING
ELSE

       IF(COUNT.EQ.0)THEN
        C0=.TRUE.
        GOTO 25291
       ENDIF
       SUMAIR=SUMAIR/FLOAT(COUNT)
       SUMSIN=SUMSIN/FLOAT(COUNT)  !COUNT INCLUDES ZEROED DIRECTIONS
8/5/97
       SUMCOS=SUMCOS/FLOAT(COUNT)
       SUMSPEED=SUMSPEED/FLOAT(COUNT)

C8/5/97 WEATH5 MODS, NOTE THAT FOR SPEEDS LESS THAN DIRECTION THRESHOLD
C THE SPEED IS SET TO 0 TEMPORARILY IN CALCULATING SUMSIN, SO THAT THIS
C MEASUREMENT DOES NOT CONTRIBUTE TO THE CALCULATION OF WIND DIRECTION
C IT DOES NOT AFFECT THE WIND SPEED SCALER AVERAGE, HOWEVER,
C BECAUSE OF THE POSSIBLE SETTING OF TMPSPD = 0, WHEN SPEED LT DIRTHR,
C COUNT MAY BE LARGER THAN THE ACTUAL NUMBER OF MEASUREMENTS USED TO
CALCULATE
C AVERAGE OF SUMSIN AND SUMCOS, HOWEVER, SINCE THAT IS ONLY USED TO
CALCULATE
C THE ARCTAN (RATIO OF THE TWO) IT DOESN'T MATTER IF COUNT IS OFF,
BECUASE
C IT CANCELS OUT

C NOTE, THIS IS SAME LINE AS IN WEATH3, BUT ANGLE NOW REFLECTS
C SPEED-WEIGHTED AVERAGE
C
C 2/17/93 MUST CHECK FOR POSSIBILITY THAT SUMSIN,SUMCOS ARE BOTH 0,
WHICH
C CAN HAPPEN WHEN WIND SPEEDS ARE 0 SINCE THIS IS NOW WEIGHTED AVG
C WEIGHTED BY SPEED
```

```
       IF(SUMSIN.EQ.0..AND.SUMCOS.EQ.0.)THEN
        ANGLE=-999999.
       ELSE
        ANGLE=360.*ATAN2(SUMSIN,SUMCOS)/6.2832
       ENDIF

C FOLLOWING 2 LINES ADDED FOR WEATHA, STANA REQUIRES POSITIVE
C ANGLES AND ANGLE MAY AT THIS POINT BE NEGATIVE
       TMPANG=ANGLE
       IF(TMPANG.LT.0.)TMPANG=360.+TMPANG

C FOLLOWING CODE FOR WEATH4 TO CALCULATE STANDARD DEVIATION
C (SEE CIMIS MANUAL FOR FAXED ALGORITHMS DERIVATION)
C FIRST CALCULATE MAGNITUDE OF VECTOR SUM

       DUMM01=SUMCOS*SUMCOS+SUMSIN*SUMSIN
c i don't think this is even used anymore in output, because decided
c some time ago not to calculate stabilities, which is what sd of
c wind dir was used for
        IF(DUMM01.LT.0.)THEN     !check for negative value 3/2/01
          WRITE(2,8911)DUMM01
8911      FORMAT(1X,'NEGATIVE VALUE FOR MAGNITUDE: SETTING =0.')
          DUMM01=0.
        ENDIF
C      MAGVEC=SQRT(SUMCOS*SUMCOS+SUMSIN*SUMSIN)
       MAGVEC=SQRT(DUMM01)

C MAGVEC MAY NO LONGER BE ACCURATE 8/5/97
C BECAUSE COUNT MAY NOT REFLECT ACTUAL NUMBER OF MEASUREMENTS USED TO
DETERMINE
C SUMCOS AND SUMSIN, SEE DISCUSSION ABOVE

C NOW CALCULATE STANDARD DEVIATION
       IF (SUMSPEED.LE.0.)THEN             !added more checks 3/5/01
        SD=-999999.
       ELSEIF (MAGVEC.LE.0.)THEN
        SD=-999999.
       ELSEIF (MAGVEC.GT.SUMSPEED) THEN
        SD=-999999.
        WRITE(2,1781)MAGVEC,SUMSPEED
1781    FORMAT(1X,'WARNING: MAGNVEC > SUMSPEED ',2E14.6)
       ELSEIF (MAGVEC.EQ.SUMSPEED) THEN
        SD=81.
        WRITE(2,8091)MAGVEC,SUMSPEED
8091    FORMAT(1X,'WARNING: MAGVEC=SUMSPEED ',2E14.6)
       ELSE
        SD=81.*SQRT(1.-(MAGVEC/SUMSPEED))
       ENDIF             !hopefully 3/5/01 this will take care of
problems

C.................FOLLOWING WEATH3 LINES REPLACED BY ABOVE.......
C FOLLOWING LINES ADDED FOR WEATH3, ANGLE IS MEAN ANGLE, THERE ARE
C COUNT ANGLES IN ARRANG

C      SD=STANA(ARRANG,COUNT,TMPANG)
C.................................................................
```

```
C FOLLOWING LINE DETERMINES IF THIS SET OF MEASUREMENTS MOSTLY IN
C THE DAYTIME OR MOSTLY AT NIGHT ACCORDING TO USER PROVIDED
C SUNRISE, SUNSET VALUES

      IF(FLOAT(DAYCNT)/FLOAT(COUNT).LT.0.5)THEN
       NIGHT=.TRUE.
       DN='NIGHT'
      ELSE
       NIGHT=.FALSE.
       DN='DAY  '
      ENDIF

C NOW DETERMINE STABILITY CLASS

      CLASS=STAB(SD,NIGHT,SUMSPEED)

C FOLLOWINGLINE CHANGES MEAN WIND FROM 'FROM' TO 'TO' DIRECTION
NECESSARY
C FOR ISCST
       ANGLE=ANGLE+180
25291  IF(C0)THEN
        WRITE(2,782)INT(SMONTH),INT(SDAY),INT(SHOUR),INT(SMIN),
     1INT(EMONTH),INT(EDAY),INT(EHOUR),INT(EMIN)
        WRITE(6,782)INT(SMONTH),INT(SDAY),INT(SHOUR),INT(SMIN),
     1INT(EMONTH),INT(EDAY),INT(EHOUR),INT(EMIN)
782     FORMAT(I2,'/',I2,I3,':',I2,' TO ',I2,'/',I2,I3,':',I2,
     1' NO DATA RECORDS')
        C0=.FALSE.
        GOTO9765
       ENDIF
       WRITE(2,500)INT(SMONTH),INT(SDAY),INT(SHOUR),INT(SMIN),
     1INT(EMONTH),INT(EDAY),INT(EHOUR),INT(EMIN),SUMAIR,ANGLE,
c     2MAGVEC,   COUNT,SD,CLASS,DN    8/5/97 this is weath4 line
     2sumspeed,   COUNT,DN    !8/5/97 weath5
C     2SUMSPEED,COUNT,SD,CLASS,DN          !THIS IS WEATH3 LINE

500    FORMAT(I2,'/',I2,I3,':',I2,' TO ',I2,'/',I2,I3,':',I2,3F10.3,5X,
c     1'(',I3,')',2X,F9.3,3X,I2,3X,A5) !weath4 line
     1'(',I3,')',2X,           3X,A5) !weath5 8/5/97

C     write(2,78001)sumspeed,magvec
C78001      format(1x,'debugging: sumspeed,magvec ',2f10.5)
       CALL M2A
       WRITE(6,500)INT(SMONTH),INT(SDAY),INT(SHOUR),INT(SMIN),
     1INT(EMONTH),INT(EDAY),INT(EHOUR),INT(EMIN),SUMAIR,ANGLE,
c    2MAGVEC,   COUNT,SD,CLASS,DN !weath4
     2sumspeed,   COUNT,DN  !weath5 8/5/97

C FOLLOWING LINE CALLS ROUTINE TO WRITE OUT LINE TO ISCST3 MET DATA
FILE
C AND INCREMENT HOUR COUNTER AND DATE STUFF 8/12/97
      CALL MAK3DO(YY,MM,DD,HH,ANGLE,SUMSPEED,SUMAIR,MIX,MIX)

C      write(2,510)sumsin,sumcos
C510   format(1x,'sumsin,sumcos ',2f15.5)
       WRITE(6,524)COUNT
524    FORMAT(51X,I4,' RECORDS PROCESSED')
```

```
9765   IF (ENDOF) GOTO1001
       SUMAIR=0.
       SUMSIN=0.
       SUMCOS=0.
       SUMSPEED=0.

C REINITIALIZE ARRANG TO NEGATIVES, THEN PROGRAM WILL BOMB
C IN STANA IF COUNT GETS MESSED UP BY BEING TOO SMALL
       DO 527 LK=1,COUNT
527    ARRANG(LK)=-1.

       COUNT=0.
       DAYCNT=0
       ENGAG=.TRUE.
       JUSTPR=.TRUE.
       GOTO10
      ENDIF
      STOP

C END OF FILE REACHED ON LINE READ

1000  CONTINUE
      ENDOF=.TRUE.
      GOTO504
1001  WRITE(6,1500)
1500  FORMAT(1X,'END OF INPUT FILE REACHED, STOPPING ')
      CLOSE(1)
      CLOSE(2)
      STOP

C ERROR ON LINE READ

2000  CONTINUE
      WRITE(6,2500)LINE(1:80)
2500  FORMAT(1X,'ERROR TRYING TO READ LINE, LAST LINE WAS',/1X,A80)
      CLOSE(1)
      CLOSE(2)
      STOP

C END OF FILE ON PARSING?

3000  CONTINUE
      WRITE(6,3500)LINE(1:80)
3500  FORMAT(1X,'EOF ON PARSE?! ',/1X,A80)
      CLOSE(1)
      CLOSE(2)
      STOP

C ERROR ON PARSING

4000  CONTINUE
      WRITE(6,4500)LINE(1:80)
4500  FORMAT(1X,'ERROR ON PARSE ',/1X,A80)
      CLOSE(1)
      CLOSE(2)
      STOP
      END
```

```
C     Last change:  BJ    5 Dec 2009    1:49 pm
      LOGICAL FUNCTION ASK(MESS,xLEN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C!BRJ091206 updated THE NEW WAY TO CALL THIS IS
C          IF(ask(mess,LEN_TRIM(mess)))then
C WITH MESS CONTAINING THE MESSAGE (UP TO MAXSTRLEN CHARS)
C AND LEN_TRIM SENDING THE LENGHT TO THE FUNCTION ASK
C THESE ARE DECLARED AS INTENT IN VARAIBLES
C THE NEW STRING STANDARD WILL BE 200 FOR THESE GENERIC STRINGS
C
C
C  TAKES MESSAGE IN ARRAY MESS OF LENGTH N AND ASKS YES NO QUESTION
C  OF FORM --ASK QUESTION?(Y/N)
C  THEN IT READS IN Y/N ANSWER AND SETS ITSELF TRUE OR FALSE
C
      IMPLICIT INTEGER (A-Z)
C       BYTE MESS(50),FORM(30),ANS
C       ENCODE (30,100,FORM)LEN
C100        FORMAT ('(''$--'',',I2,'A1,''?(Y/N) '')')
CD        WRITE (5,200)LEN,FORM
CD200        FORMAT (1X,'LEN=',I2,'FORM= ',30A1)
C
C PREVIOUS COMMENTED LINES LEFT IN FOR HISTORICAL PURPOSES
C AS THEY WAY IT WAS WITH ENCODE STATEMENTS ON THE PDP
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
CCCC
        PARAMETER(MAXSTRLEN=200)
      CHARACTER*(MAXSTRLEN) MESS
      CHARACTER*30 FORM
      CHARACTER*1 ANS
        INTENT (IN) MESS,XLEN
C
C CREATE APPROPRIATE FORMAT
      WRITE (FORM,100)xLEN
100     FORMAT('(1X,''--'',A',I3,',''?(Y/N) ''\)')
C     write (0,150)form
C150     format(a30)
2       CONTINUE
      WRITE (6,FORM)MESS(1:xLEN)
      READ (5,350)ANS
350        FORMAT (A1)
      IF (ANS.EQ.'Y'.OR.ANS.EQ.'y')ASK=.TRUE.
      IF (ANS.EQ.'N'.OR.ANS.EQ.'n')ASK=.FALSE.
      IF (ANS.EQ.'N'.OR.ANS.EQ.'Y'.OR.ANS.EQ.'n'.or.ANS.EQ.'y')RETURN
      WRITE (6,400)
400        FORMAT(1X,'TYPE A "Y" OR AN "N" TO ANSWER QUESTION ')
      GO TO 2
      END
```

```
C     Last change:  BJ    5 Dec 2009    1:20 pm
      SUBROUTINE bjPACK(S,XLEN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C!BRJ091201 UPDATED
c
c 8/4/97 pack conflicts with fortran intrinsic called pack
c renamed to bjpack
C
C PACK REMOVES ALL EMBEDDED AND LEADING BLANKS IN CHARACTER
C STRING S AND RETURNS S LEFT JUSTIFIED AND PACKED
C LEN SHOULD BE LENGTH OF S
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
        PARAMETER(STRMAXLEN=200)
      CHARACTER*(STRMAXLEN) S
        INTENT (IN OUT) S
        INTENT (IN ) XLEN
C
      INCREM=0
      LEN1=LEN_TRIM(S)

        IF(LEN1.GT.STRMAXLEN)THEN
          WRITE(6,8989)LEN1,STRMAXLEN
8989      FORMAT(1X,'ERROR FROM BJPACK: STRINGTOOBIG ',2I5)
          STOP
        ENDIF

      DO 10 I=1,LEN1
        IF(I+INCREM.GT.LEN1) GOTO30
        IF(S(I+INCREM:I+INCREM).NE.' ')THEN
          S(I:I)=S(I+INCREM:I+INCREM)
        ELSE
C
C WE'VE GOT A BLANK, SKIP TO NEXT CHARACTER
1         CONTINUE
          INCREM=INCREM+1
          IF (I+INCREM.GT.LEN1) GOTO30
          IF (S(I+INCREM:I+INCREM).EQ.' ') GOTO1
C
C WE MUST HAVE FOUND ANOTHER NON BLANK
          S(I:I)=S(I+INCREM:I+INCREM)
        ENDIF
10    CONTINUE
C
C NOW BLANK OUT REMAINING CHARACTERS IN S
30    IF (I-1.GE.LEN1)RETURN
      DO 20 J=I,LEN1
        S(J:J)=' '
20    CONTINUE
      RETURN
      END
```

```
C     Last change:  BJ    8 Dec 2009    4:12 pm
      SUBROUTINE DLINE(UU)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C OUTPUT LINE OF 60 DASHES TO UNIT UU
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
        INTENT (IN) UU

      WRITE(UU,100)
100   FORMAT(1X, '------------------------------',
     1             '------------------------------')
      RETURN
      END
```

```
C     Last change:  BJ    5 Dec 2009    1:22 pm
      SUBROUTINE FLUSHC(STRING,XLEN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C!BRJ091206 updated
C SUBROUTINE FLUSHC SETS STRING OF LENGTH LEN ALL
C EQUAL TO BLANKS
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER (A-Z)
        PARAMETER(MAXSTRLEN=200)
      CHARACTER*(MAXSTRLEN) STRING
C
        INTENT(IN) XLEN
        INTENT(IN OUT) STRING
C
        IF(XLEN.GT.MAXSTRLEN)THEN
          WRITE(6,1818)XLEN,MAXSTRLEN
1818      FORMAT(1X,'ERROR FROM FLUSCH INPUT STR TOO BIG',2I5)
          STOP
        ENDIF

      IF (XLEN.EQ.0) THEN
        RETURN
      ELSE IF (XLEN.LT.0) THEN
        WRITE(0,200)XLEN
200     FORMAT(1X,'WARNING FROM FLUSHC: XLEN < 0: XLEN=',I4)
        RETURN
      ELSE
        DO 10 I=1,MIN(XLEN,MAXSTRLEN)
10      STRING(I:I)=' '
        RETURN
      ENDIF
      END
```

```
C     Last change:  BJ    9 Dec 2009    3:54 pm
       SUBROUTINE M1A(KK,DIRTHR)
       !MODIFIED 091209
c at this time, dirthr, (wind speed direction threshold) comes in
c as mph
      REAL DIRTHR
      CHARACTER*3 MONT(12)
      CHARACTER*8 YMD
      CHARACTER*10 HHMMSS
      CHARACTER*5 ZN
      INTEGER V(8)
C     character*1 dumb
        INTENT (IN) KK, DIRTHR
      DATA MONT/'JAN','FEB','MAR','APR','MAY','JUN','JUL',
     1'AUG','SEP','OCT','NOV','DEC'/
C
C     write(0,7878)
C7878 format(1x,'this mod of m1a made it into library ')
C     read(0,100)dumb
C100  format(a1)
C
      CALL DATE_AND_TIME(YMD,HHMMSS,ZN,V)
      IYR=V(1)
      IMON=V(2)
      IDAY=V(3)
C     CALL GETTIM(IHR,IMIN,ISEC,I100TH)
      IHR=V(5)
      IMIN=V(6)
      ISEC=V(7)
      I100TH=V(8)
      WRITE(KK,1000)MONT(IMON),IDAY,IYR,IHR,IMIN
1000  FORMAT(1X,'TODAY''S DATE: ',A3,1X,I2,1X,I4,
     1/9X,'TIME: ',I2.2,':',I2.2)
C
      WRITE(kk,1909)dirthr,0.447*dirthr
1909  FORMAT(1X,'   WEATH6A: LAST MODIFIED 12/17/2009. SCALER WIND SP',
     *'EED,',
     */1X,'      INCLUDES ALL WIND MSMNTS AND AVERAGE DIRECTI',
     *'ON BASED',
     */1X,'      ONLY ON MSMNTS WHERE SPEED> ',f6.4,'MPH',
     *1x,' (',f6.4,'M/S).')
      CALL DLINE(KK)

      RETURN
        END
```

```
       SUBROUTINE m1a_95
C
      WRITE(6,1900)
1900  FORMAT(1X,'                 WEATH6A (last mod 12/17/09)',
     */1X,' ',
     */1X,'  This program takes a met data file and summarize',
     *'s',
     */1X,'  for user specified periods the ambient air tempe',
     *'rature,',
     */1X,'  wind direction, and wind speed.  Also produces I',
     *'SCST3',
     */1X,'  usable met data file.')
      WRITE(6,2000)
2000  FORMAT(1X,' ',
     */1X,'  The input file is assumed to be in the following',
     *' form:',
     */1X,' ',
     */1X,'  mm,dd,hh,mi,int,at,agt,dir,spd where commas sepa',
     *'rate',
     */1X,'  values and',
     */1X,' ')
      WRITE(6,2100)
2100  FORMAT(1X,'  mm=month           dd=day               hh=hour',
     */1X,'  mi=minute          int=dummyinterval    at=ambie',
     *'nt temp (F)',
     */1X,'  agt=dummyholder    dir=FROM wind dir    spd=wind',
     *' spd (mph)',
     */1X,' ',
     */1X,'  Dummy fields not used, but must be present.',
     */1X,' ')
        RETURN
        END
```

```
       SUBROUTINE m1b_95
C
      WRITE(6,1900)
1900  FORMAT(1X,'  mm,dd,hh,mi,int,at,agt,dir,spd where commas sepa',
     *'rate',
     */1X,'  values and',
     */1X,' ',
     */1X,'  mm=month           dd=day               hh=hour',
     */1X,'  mi=minute          int=dummyinterval    at=ambie',
     *'nt temp (F)',
     */1X,'  agt=dummyholder    dir=FROM wind dir    spd=wind',
     *' spd (mph)')
      WRITE(6,2000)
2000  FORMAT(1X,' ',
     */1X,'  Dummy fields not used, but must be present.',
     */1X,' ',
     */1X,'  Here are two example lines: first line is June 2',
     *'3, 1400 hours',
     */1X,'  20 minutes, 74.53 F, wind from 216.4 degrees (fr',
     *'om southwest)',
     */1X,'  at 12.72 miles per hour.')
      WRITE(6,2100)
2100  FORMAT(1X,' ',
     */1X,'  6,23,14,20,1,74.53,1,216.4,12.72',
     */1X,'  6,23,14,22,1,73.56,1,221,11.12',
     */1X,' ',
     */1X,'  Data lines must be sorted with oldest data first',
     *'.',
     */1X,' ',
     */1X,'  WEATH6A produces output file giving time interva',
     *'l, average')
      WRITE(6,2200)
2200  FORMAT(1X,'  ambient air temperature in degrees Kelvin, avera',
     *'ge wind',
     */1X,'  direction TO ( mod(180+FROM,360) ), scaler avera',
     *'ge wind speed',
     */1X,'  (meters/second=.447*milesperhour).',
     */1X,' ')
        RETURN
        END
```

```
       SUBROUTINE M2
C
      WRITE(2,1909)
1909  FORMAT(1X,' ',
     */3X,'                          SCA AVG    180+AVG  SCA ',
     *'AVG    NUMBER OF',
     */1X,'.......TIME PERIOD........  TEMPERA K  DIRECTI  SPD ',
     *'M/S    RECORDS    N/D')
        RETURN
        END
```

```
C     Last change:  BJ    9 Dec 2009    3:55 pm
       SUBROUTINE M2A
C
      WRITE(6,1909)
1909  FORMAT(1X,' ',
     */3X,'                          SCA AVG    180+AVG  AVG ',
     *'SPEED  NUMBER OF',
     */1X,'.......TIME PERIOD........  TEMPERA K  DIRECTI    M/',
     *'S      RECORDS   N/D')
        RETURN
        END
```

```
C     Last change:  BJ    9 Dec 2009    3:58 pm
      SUBROUTINE MAK3DO(YY,MM,DD,HH,FLOW,WSPD,TEMP,RMIX,UMIX)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THIS SUBROUTINE WRITES OUT CURRENT VALUES FOR YEAR,MONTH,DAY,
C  HOUR,FLOW (TO DIRECTION), SPEED, TEMPERATURE, RURAL, URBAN MIXING HT
C  THEN UPDATES YY,MM,DD,HH TO NEXT HOUR (UPDAT MAKES APPROPRIATE
C  CHANGES IF NECESSARY TO YEAR, MONTH, DAY
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL FLOW,WSPD,TEMP,RMIX,UMIX
        INTENT (IN) FLOW,WSPD,TEMP,RMIX,UMIX
        INTENT (IN OUT) YY, MM, DD, HH

      WRITE(3,100)YY,MM,DD,HH,FLOW,WSPD,TEMP,' X',RMIX,UMIX
100   FORMAT(4I2,2F9.4,F6.1,A2,2F7.1)

      HH=HH+1
      CALL UPDAT(YY,MM,DD,HH)
      RETURN
      END
```

```
C     Last change:  BJ   14 Dec 2009   11:04 am
      SUBROUTINE MAK3IN(YY,MM,DD,HH,MIX)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C GET STARTING VALUES FOR YEAR,MONTH, DAY, HOUR, MIXING HEIGHT
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL MIX
        INTENT (OUT) YY,MM,DD,HH,MIX

50    WRITE(6,100)
100   FORMAT(1X,'ENTER STARTING VALUES FOR YEAR,MONTH,DAY,HOUR',
     1/1X,'FOR FIRST RECORD OF ISCST3 MET CONTROL FILE ',
     1/1X,'THE FORMAT IS YY,MM,DD,HH (FOR EXAMPLE, 90,3,25,15) ')
      READ(5,*,ERR=50)YY,MM,DD,HH

C180  FORMAT(4I8)

190   WRITE(6,200)
200   FORMAT(1X,'NOW ENTER MIXING HEIGHT IN METERS ')
      READ(5,*,ERR=190)MIX
C250  FORMAT(F10.0)
      RETURN
      END
```

```
C     Last change:  BJ    9 Dec 2009    4:10 pm
      LOGICAL FUNCTION NDET(NHOUR,NMIN,RHOUR,RMIN,SHOUR,SMIN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C THIS LOGICAL FUNCTION RETURNS TRUE IF
C NIGHTTIME DEFINED AS 1 HOUR B4 SUNRISE
C TO 1 HR AFTER SUNSET
C
C USER MUST WORRY ABOUT PACIFIC DAYLIGHT VS STANDARD TIME
C
C NHOUR,NMIN: USER SUPPLIED HOUR (MILITARY) AND MINUTE OF CURRENT TIME
C RHOUR,RMIN: USER SUPPLIED HOUR/MIN OF SUNRISE
C SHOUR,SMIN: USER SUPPLIED HOUR/MIN OF SUNSET
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL NHOUR,NMIN,RHOUR,RMIN,SHOUR,SMIN
      REAL EARLY,LATE,CURREN,MINCON
      LOGICAL CK
        INTENT (IN) NHOUR,NMIN,RHOUR,RMIN,SHOUR,SMIN

      NDET=.FALSE.

C CHECK FOR REASONABLENESS OF VALUES

      IF(.NOT.CK(NHOUR,NMIN))THEN
       WRITE(6,500)NHOUR,NMIN
500    FORMAT(1X,'NDET: BAD VALUES CURRENT HOUR ',2F10.2)
       STOP
      ENDIF
      IF(.NOT.CK(RHOUR,RMIN))THEN
       WRITE(6,501)RHOUR,RMIN
501    FORMAT(1X,'NDET: BAD VALUES SUNRISE HOUR ',2F10.2)
       STOP
      ENDIF
      IF(.NOT.CK(SHOUR,SMIN))THEN
       WRITE(6,502)SHOUR,SMIN
502    FORMAT(1X,'NDET: BAD VALUES SUNSET HOUR ',2F10.2)
       STOP
      ENDIF
      IF(RHOUR.GE.SHOUR)THEN
       WRITE(6,503)RHOUR,SHOUR
503    FORMAT(1X,'NDET: BAD VALUES SUNRISE/SET HOURS ',2F10.2)
       STOP
      ENDIF
      IF(RHOUR.LT.4..OR.RHOUR.GT.9.)THEN
       WRITE(6,504)RHOUR
504    FORMAT(1X,'NDET: UNREASONABLE SUNRISE HOUR ',F10.2)
       STOP
      ENDIF
      IF(SHOUR.LT.16..OR.SHOUR.GT.21.)THEN
       WRITE(6,505)SHOUR
505    FORMAT(1X,'NDET: UNREASONABLE SUNSET HOUR ',F10.2)
       STOP
      ENDIF
```

```
C DETERMINE PORTION OF DAY FOR EACH INSTANCE IN MINUTES FROM MIDNIGHT

      EARLY=60.+MINCON(RHOUR,RMIN)
      LATE=-60.+MINCON(SHOUR,SMIN)
      CURREN=MINCON(NHOUR,NMIN)
C     write(0,1000)early,late,curren
C1000 format(1x,'early,late,curren',3f10.2)
      IF(EARLY.LE.CURREN.AND.CURREN.LE.LATE)THEN
       NDET=.FALSE.
      ELSE
       NDET=.TRUE.
      ENDIF
      RETURN
      END
```

```
C     Last change:  BJ    9 Dec 2009    3:00 pm
      logical function of(iu,f2)
cccccccccccccccccccccccccccccccc
c
c opens a file, allows user to overwrite existing if they want to
c
cccccccccccccccccccccccccccccccc
      implicit integer(a-z)
      character*200 mess
      character*200 f2
      logical ask,THERE
        INTENT (IN) iu
        INTENT (IN OUT) f2


      of=.false.
1     write(6,577)
577   format(1x,'enter filename ')
      read(5,50)f2
50    format(a200)
      open(unit=iu,status='new',file=f2,iostat=ierr,
     1       err=1000)
1000  if(ierr.gt.0)then
          INQUIRE(FILE=F2,IOSTAT=IERR2,EXIST=THERE)
          !MOST LIKELY THE ERROR IS BECAUSE THEFILE EXISTS
          IF(THERE)THEN
            call flushc(mess,200)
            mess(1:24)='Overwrite existing file '
          if(ask(mess,24))then
            open(unit=iu,status='replace',file=F2,iostat=ierr,
     1             err=1000)
            if(ierr.le.0)then
              goto 400
            else
              write(6,100)f2(1:50),ierr
              call iostat_msg(ierr,mess)
              write(6,120)mess(1:79)
                call flushc(mess,200)
                mess(1:21)='try another filename '
              if (ask(mess,21))then
                goto1
              else
                return
              endif
            endif
          else
            goto1
          endif
        else
          write(6,100)f2(1:50),ierr
100       format(1x,'an error trying to open your file (',a50,')',
     1            ' occurred and its error number ',i10)
          call iostat_msg(ierr,mess)
          write(0,120)mess(1:79)
120       format(1x,a79)
            call flushc(mess,200)
            mess(1:20)='try another filename '
```

```
          if (ask(mess,20))then
            goto1
          else
            return
          endif
        endif
      endif
400   continue
      of=.true.
      return
      end
```

```
C     Last change:  BJ   14 Dec 2009   10:58 am
      LOGICAL FUNCTION OPENE(UNITNO,FNAME,NLEN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C !BRJ091201 LEN IS NOW AN INTRINSIC, MUST CHANGE ALL REFS TO IT
C !BRJ091201 WILL CHANGE TO XLEN
C LOGICAL FUNCTION OPENE ATTEMPTS TO OPEN AN EXISTING FILE
C CHECKS FOR NON BLANK FNAME
C IF NONBLANK THEN
C ATTEMPTS TO OPEN AN EXISTING FILE CALLED 'FNAME'.
C IF BLANK THEN ASKS FOR FILENAME
C IF OPEN IS UNSUCCESSFUL USER GETS ANOTHER OPPORTUNITY
C IF OPEN IS NOT ACCOMPLISHED THEN
C OPENE IS FALSE, IF SUCCESSFUL OPENE IS TRUE
C NLEN IS RETURNED LENGTH OF FILENAME
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER (A-Z)
      LOGICAL ASK,THERE,NULLS
        PARAMETER(MAXSTRLEN=200)
      CHARACTER*200 FNAME,MESS
        INTENT(IN OUT)FNAME, NLEN
        INTENT (IN) UNITNO !THIS IS UNIT# TO OPEN
      CALL FLUSHC(FNAME,MAXSTRLEN)
C     WRITE (0,6000)FNAME
C6000 FORMAT(1X,'FIRST LINE IN OPENE ',A40)
C
1       CONTINUE
C
C FIRST CASE IS WHERE NULLS=TRUE, IE FNAME IS BLANK AT THIS POINT
C
C     XLEN=SLENG(FNAME,MAXSTRLEN)
        XLEN=LEN_TRIM(FNAME)
      NLEN=XLEN
C     WRITE (0,5590)FNAME
C5590 FORMAT(1X,'JUST AFTER SLENG ',A30)
C     IF (ICHAR(FNAME(LEN:LEN)).NE.32) FNAME(LEN+1:LEN+1)=CHAR(32)
C     WRITE (0,9000)LEN
C9000 FORMAT(1X,'LEN= ',I3)
      IF (NULLS(FNAME,NLEN))THEN
       WRITE (6,100)
100      FORMAT(1X,'--ENTER NAME OF FILE TO OPEN '\)
       READ (5,200,IOSTAT=RESTAT)FNAME
200      FORMAT(A200)
C      XLEN=SLENG(FNAME,MAXSTRLEN)
         XLEN=LEN_TRIM(FNAME)
         NLEN=XLEN
       IF (RESTAT)2000,20,1000
C
C FIND OUT IF FILE IS THERE
20     IF (NULLS(FNAME,MAXSTRLEN))THEN
         WRITE(6,250)
250        FORMAT(1X,'--FILENAME IS BLANK! ')
C
C        IF IN THIS SECTION, USER HAS ENTERED BLANK FILEFNAME
           CALL FLUSHC(MESS,200)
           MESS(1:42)='DO YOU WANT TO ENTER A DIFFERENT FILENAME '
         IF(ASK('DO YOU WANT TO ENTER A DIFFERENT FILENAME ',42))
```

```
     1     THEN
           CALL FLUSHC(FNAME,MAXSTRLEN)
           GOTO1
          ELSE
           OPENE=.FALSE.
           NLEN=0
           RETURN
          ENDIF
        ELSE
         CONTINUE
        ENDIF
       ENDIF
C
C     WRITE (0,7889)FNAME
C7889 FORMAT(1X,'JUST BEFORE INQUIRE: FNAME IS ',A30)
C     DO 7990 IO=1,16
C7990 WRITE (0,7991)IO,FNAME(IO:IO),ICHAR(FNAME(IO:IO))
C7991 FORMAT(1X,I3,1X,A1,1X,I4)
      INQUIRE (FILE=FNAME,EXIST=THERE,IOSTAT=IOCHK)
C
      IF (.NOT.THERE) THEN
        WRITE (6,300)FNAME
300       FORMAT(1X,'FOLLOWING FILE DOES NOT EXIST: ',A200)
          CALL FLUSHC(MESS,200)
          MESS(1:12)='TRY ANOTHER '
        IF (ASK(MESS,12)) THEN
          CALL FLUSHC(FNAME,MAXSTRLEN)
          GOTO1
        ELSE
          OPENE=.FALSE.
          RETURN
        ENDIF
      ELSE IF(IOCHK.NE.0) THEN
        WRITE(6,377)IOCHK,FNAME
377     FORMAT(1X,'ERROR ON FILE ',A40,' ERR= ',I5)
          CALL FLUSHC(MESS,200)
          MESS(1:12)='TRY ANOTHER '
        IF (ASK(MESS,12)) THEN
          CALL FLUSHC(FNAME,MAXSTRLEN)
          GOTO1
        ELSE
          OPENE=.FALSE.
            NLEN=0
          RETURN
        ENDIF
      ELSE
C     WRITE (0,8889)FNAME
C8889 FORMAT(1X,'JUST BEFORE OPEN: FNAME IS ',A30)
C     DO 8990 IO=1,30
C8990 WRITE (0,8991)IO,FNAME(IO:IO),ICHAR(FNAME(IO:IO))
C8991 FORMAT(1X,I3,1X,A1,1X,I4)
        OPEN (UNIT=UNITNO,FILE=FNAME,STATUS='OLD')
        OPENE=.TRUE.
        RETURN
      ENDIF
C
1000    CONTINUE
```

```
        CALL FLUSHC(MESS,200)
        MESS(1:35)='ERROR ON FILENAME INPUT, TRY AGAIN '
      IF(ASK(MESS,35))THEN
        CALL FLUSHC(FNAME,MAXSTRLEN)
        GOTO1
      ELSE
       OPENE=.FALSE.
      ENDIF
      RETURN
2000    WRITE (6,2005)
2005    FORMAT(1X,'USER CNTRL Z - QUIT ')
      STOP
      END
```

```
C     Last change:  BJ   16 Dec 2009    4:16 pm
        LOGICAL FUNCTION OPENN(UNITNO,FNAME,NLEN)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C UPDATED 091205BRJ  REMOVED REFERENCE TO "LEN" WHICH IS NOW INTRINSIC
C
C LOGICAL FUNCTION OPENN ATTEMPTS TO OPEN A NEW FILE
C CHECKS FOR NON BLANK FNAME
C IF NONBLANK THEN
C ATTEMPTS TO OPEN A NEW FILE CALLED 'FNAME'.
C CHECKS TO SEE IF 'FNAME' ALREADY EXISTS TO AVOID OVERWRITE
C IF BLANK THEN ASKS FOR FILENAME
C IF OPEN IS UNSUCCESSFUL USER GETS ANOTHER OPPORTUNITY
C IF OPEN IS NOT ACCOMPLISHED THEN
C OPENN IS FALSE, IF SUCCESSFUL OPENN IS TRUE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER (A-Z)
      LOGICAL ASK,THERE,NULLS
        PARAMETER(maxstrlen=200)
      CHARACTER*(maxstrlen) FNAME,MESS
        INTENT (IN OUT) fname,nlen
        INTENT (IN) unitno
C
C     CALL CLEANS (FNAME,1,maxstrlen)
1       CONTINUE
C
C CHECK FOR BLANK FILENAME
C
      aLEN=LEN_TRIM(fname)
      NLEN=aLEN
C
C FIND OUT IF INCOMING STRING ALREADY HAS A NAME IN IT
      IF (NULLS(FNAME,LEN_TRIM(FNAME)))THEN
       WRITE (6,100)
100      FORMAT(1X,'--ENTER NAME OF FILE TO OPEN '\)
c         call paus ()
       READ (5,200,IOSTAT=RESTAT)FNAME
200      FORMAT(A200)
c         write(6,2323)restat
c2323     FORMAT(1x,'restat= 'i8)
       ALEN=LEN_TRIM(FNAME)
       NLEN=ALEN
       IF (RESTAT)2000,20,1000
C
C FIND OUT IF FILE IS THERE
20     IF (NULLS(FNAME,ALEN))THEN
         WRITE(6,250)
250        FORMAT(1X,'--FILENAME IS BLANK! ')
C
C        IF IN THIS SECTION, USER HAS ENTERED BLANK FILEFNAME
           CALL FLUSHC(MESS,MAXSTRLEN)
           MESS(1:42)='DO YOU WANT TO ENTER A DIFFERENT FILENAME '
         IF(ASK(MESS,42))THEN
          CALL FLUSHC(FNAME,LEN_TRIM(FNAME))
          GOTO1
         ELSE
          OPENN=.FALSE.
```

```
          NLEN=0
          RETURN
         ENDIF
       ELSE
        CONTINUE
       ENDIF
      ENDIF
C
C
C ********************************
       INQUIRE (FILE=FNAME,EXIST=THERE,IOSTAT=FSTAT)
C ********************************
C
       IF (THERE) THEN
          WRITE (6,300)FNAME
300         FORMAT(1X,'FOLLOWING FILE EXISTS: ',A200)
            CALL FLUSHC(MESS,MAXSTRLEN)
            MESS(1:13)='OVERWRITE IT '
          IF (.NOT.ASK(MESS,13)) THEN
C           DON'T OVERWRITE FILE
            CALL FLUSHC(FNAME,LEN_TRIM(FNAME))
            GOTO1
          ELSE
C           DO OVERWRITE FILE, BUT MUST DELETE IT
            OPEN(UNIT=UNITNO,FILE=FNAME,STATUS='OLD')
            CLOSE(UNIT=UNITNO,STATUS='DELETE')
              OPEN (UNIT=UNITNO,FILE=FNAME,STATUS='NEW',IOSTAT=FSTAT)
C           VARIABLE FSTAT IS POSITIVE IF SOME ERROR OCCURS
C             MOSTLIKELY SUCH AN ERROR WOULD BE DUE TO A FAULTY FILENAM
            IF (FSTAT.GT.0)GOTO1000
            OPENN=.TRUE.
            RETURN
          ENDIF
       ELSE
C         FILE IS NOT THERE GO AHEAD AND OPEN IT
            OPEN (UNIT=UNITNO,FILE=FNAME,STATUS='NEW',IOSTAT=FSTAT)
          IF (FSTAT.GT.0)GOTO1000
          OPENN=.TRUE.
          RETURN
       ENDIF
C
1000    CONTINUE
        CALL FLUSHC(MESS,MAXSTRLEN)
        MESS(1:35)='ERROR ON FILENAME INPUT, TRY AGAIN '
      IF(ASK(MESS,35))THEN
        CALL FLUSHC(FNAME,LEN_TRIM(FNAME))
        GOTO1
      ELSE
       OPENN=.FALSE.
      ENDIF
      RETURN
2000    WRITE (6,2005)
2005    FORMAT(1X,'USER CNTRL Z - QUIT ')
      STOP
      END
```

```
C     Last change:  BJ    5 Dec 2009    1:24 pm
      SUBROUTINE PAUS
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C !BRJ091201  updated
C ALLOW PAUSE IN PROGRAM, <RETURN> TO CONTINUE
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      CHARACTER*1 A
C
      WRITE(6,100)
100   FORMAT(/1X,'                   Press <RETURN> key to continue',
     *' '\)
      READ(5,200,END=1000,IOSTAT=IERR)A
200   FORMAT(A1)
C
1000  RETURN
      END
```

```
C     Last change:  BJ    9 Dec 2009    4:12 pm
      INTEGER FUNCTION STAB(SD,NIGHT,WS)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C 091209 WILL STUB THIS OUT SINCE DON'T DO STABILITY
C CALCULATIONS ANYMORE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C 3/5/01 MODIFIED TO INCLUDE POSSIBILITY OF NEGATIVE STANDARD
C DEVIATION WHICH THEN RETURNS NEGATIVE -1 FOR STABILITY
C
C THIS FUNCTION RETURNS STABILITY CLASS 1-6 (A-F)
C USING
C
C SD: USER SUPPLIED STANDARD DEVIATION OF WIND
C DIRECTION IN DEGREES
C
C NIGHT: LOGICAL VARIABLE NIGHT
C WHICH IS FALSE IF DAYTIME, TRUE IF NIGHTTIME
C THIS DETERMINATION MUST BE MADE ELSEWHERE
C
C WS: AVERAGE WIND SPEED IN METERS PER SECOND (M/S)
C
C C BASED ON ZANETTI PGS 148-149, TABLES 7-2 AND 7-3 FOR
C DETERMINING STABILITY CLASSES BASED ON STANDARD DEVIATION
C OF HORIZONTAL WIND ANGLE AND NIGHT VS DAYTIME PLUS
C WINDSPEED
C ZANETTI, PAOLO. 1990. AIR POLLUTION MODELING. VAN NOSTRAND
C AND REINHOLD, NEW YORK.
C
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL SD,WS
      LOGICAL NIGHT

      STAB=-1
        !STUB IT OUT
        RETURN
        !STUBBED
      IF(WS.LT.0.OR.SD.LT.0.)THEN
        WRITE(0,200)WS,SD
        WRITE(2,200)WS,SD
200     FORMAT(1X,'STAB: BAD VALUE WINDSPEED/STDDEV ',2F10.5)
        STAB=-1
        RETURN
      ENDIF
      IF (SD.GE.22.5) THEN
       STAB=1
      ELSEIF (SD.GE.17.5.AND.SD.LT.22.5)THEN
       STAB=2
      ELSEIF (SD.GE.12.5.AND.SD.LT.17.5)THEN
       STAB=3
      ELSEIF (SD.GE.7.5.AND.SD.LT.12.5)THEN
       STAB=4
      ELSEIF (SD.GE.3.8.AND.SD.LT.7.5)THEN
       STAB=5
      ELSEIF (SD.GE.0..AND.SD.LT.3.8)THEN
```

```
       STAB=6
      ELSEIF (SD.LT.0.)THEN
       WRITE(0,100)SD
100    FORMAT(1X,'BAD VALUE FOR WIND DIR STANDARD DEVIATION ',F10.5)
       STAB=-1
       RETURN
      ENDIF

C NOW MAKE NIGHTTIME ADJUSTMENT IF NECESSARY FOR WIND SPEED

      IF(.NOT.NIGHT)THEN
       RETURN
      ELSE
       IF (STAB.EQ.1)THEN
         IF(WS.LT.2.9)THEN
          STAB=6
         ELSEIF(WS.GE.2.9.AND.WS.LT.3.6)THEN
          STAB=5
         ELSEIF(WS.GE.3.6)THEN
          STAB=4
         ENDIF
       ELSEIF (STAB.EQ.2) THEN
         IF(WS.LT.2.4) THEN
          STAB=6
         ELSEIF(WS.GE.2.4.AND.WS.LT.3.0)THEN
          STAB=5
         ELSEIF (WS.GE.3.)THEN
          STAB=4
         ENDIF
       ELSEIF (STAB.EQ.3) THEN
         IF(WS.LT.2.4)THEN
          STAB=5
         ELSEIF (WS.GE.2.4)THEN
          STAB=4
         ENDIF
       ENDIF
      ENDIF
      RETURN
      END
```

```
C     Last change:  BJ   10 Dec 2009    3:58 pm

      LOGICAL FUNCTION T(MONTH,DAY,HOUR,XMIN,XTIM)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C 091210 CHANGED MIN TO XMIN TO AVOID INSTRINSC AND
C OTHER MINOR CHANGES TO USE WITH LF95
C
C ATTEMPTS TO READ MONTH,DAY,HOUR,MIN
C USES CNTL Z TO END EVERYTHING
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL MONTH,DAY,HOUR,XMIN,XTIM,DAYS
      LOGICAL ASK
        INTENT (IN OUT) MONTH, DAY, HOUR, XMIN, XTIM
        CHARACTER*200 MESS

1     T=.FALSE.
      READ(0,100,ERR=1000,END=2000)MONTH,DAY,HOUR,XMIN
100   FORMAT(4F10.0)
      T=.TRUE.

C CONVER TO MINUTES FROM START OF YEAR

      DAYS=FLOAT(DATE2JUL(INT(MONTH),INT(DAY)))
      XTIM=60.*HOUR+XMIN+24.*60.*DAYS
      RETURN

1000  CONTINUE
      WRITE(6,1100)MONTH,DAY,HOUR,XMIN
1100  FORMAT(1X,'READ ERROR, TRY AGAIN (',4F5.0,')')
      RETURN

2000    CONTINUE
        CALL FLUSHC(MESS,200)
        MESS(1:20)='WANT TO END IT NOW? '
      IF(ASK(MESS,20))THEN
       CLOSE(1)
       CLOSE(2)
       STOP
      ELSE
       GOTO1
      ENDIF
      END
```

```
C     Last change:  BJ    9 Dec 2009    4:15 pm

      REAL FUNCTION TCALC(MONTH,DAY,HOUR,XMIN)

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C 091209 NOTE MIN IS INTRINSIC, CHANGE THE NAME
C
C CALCULATES MINUTES FROM START OF YEAR
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL MONTH,DAY,HOUR,XMIN,XTIM,DAYS
      INTENT (IN) MONTH,DAY,HOUR,XMIN

C CONVER TO MINUTES FROM START OF YEAR

      DAYS=FLOAT(DATE2JUL(INT(MONTH),INT(DAY)))
      TCALC=60.*HOUR+XMIN+24.*60.*DAYS
      RETURN
      END
```

```
C     Last change:  BJ   14 Dec 2009    1:14 pm
      LOGICAL FUNCTION TW4(MONTH,DAY,HOUR,XMIN,XTIM,IUNIT)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C 091212 MIN IS INSTRINSIC, RENAME IT
C 091212 UNIT IS INSTRINSIC, RENAME IT
C2/1/93
C TW4 IS FOR WEATH4 AN IS SAME AS T EXCEPT ALLOWS
C FOR READ FROM UNIT BESIDES 0
C COMPUTES NUMBER OF MINUTES FROM START OF YEAR
C
C
C ATTEMPTS TO READ MONTH,DAY,HOUR,MIN
C USES CNTL Z TO END EVERYTHING
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT INTEGER(A-Z)
      REAL MONTH,DAY,HOUR,XMIN,XTIM,DAYS
      LOGICAL ASK

        PARAMETER(maxstrlen=200)
        CHARACTER*(MAXSTRLEN) MESS
        INTENT (IN) IUNIT
        INTENT (OUT) MONTH,DAY,HOUR,XMIN,XTIM

1     TW4=.FALSE.
      READ(IUNIT,*,ERR=1000,END=2000)MONTH,DAY,HOUR,XMIN
C100  FORMAT(4F10.0)
      TW4=.TRUE.

C CONVER TO MINUTES FROM START OF YEAR

      DAYS=FLOAT(DATE2JUL(INT(MONTH),INT(DAY)))
      XTIM=60.*HOUR+XMIN+24.*60.*DAYS
      RETURN

1000  CONTINUE
      WRITE(6,1100)MONTH,DAY,HOUR,XMIN
1100  FORMAT(1X,'READ ERROR FROM TW4, (',4F5.0,')')
      RETURN

2000    CALL FLUSHC(MESS,200)
        MESS(1:20)='WANT TO END IT NOW? '
      IF(ASK(MESS,20))THEN
       CLOSE(1)
       CLOSE(2)
       STOP
      ELSE
       GOTO1
      ENDIF
      END
```