PREDICTING AIR TEMPERATURES IN CITY STREETS ON THE BASIS OF MEASURED REFERENCE DATA

A thesis submitted to The University of Adelaide in fulfilment of the requirements for a degree of Doctor of Philosophy

Evyatar Erell

The School of Architecture, Landscape Architecture and Urban Design

February 2005

APPENDICES

Appendix A: Calculation of view factors.

Appendix B: Calculation of shaded areas.

Appendix C: CAT code (FORTRAN)

APPENDIX A: CALCULATION OF VIEW FACTORS

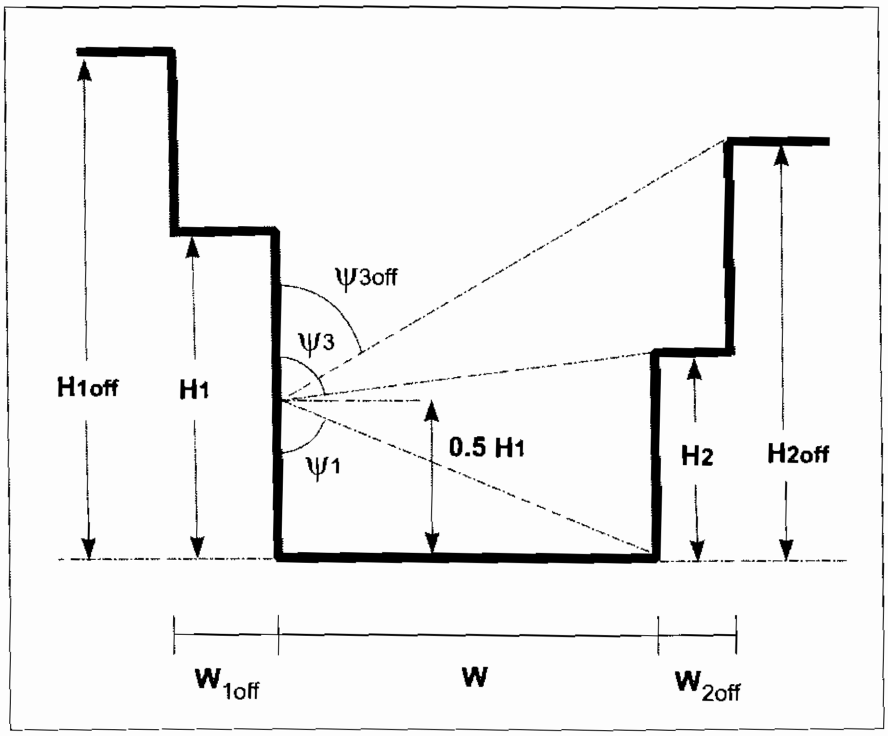

Figure 1: View factors from the middle of canyon wall 1

Ψ_1=ATAN(W/(H_1*0.5))

Ψ_3=ATAN(W/(H_2-0.5*H_1)), but Ψ_3 cannot exceed $\pi/2$.

Ψ_{3off}=ATAN((W+W_{2off})/(H_{2off}-(0.5*H_1)))

VFw1_g=0.5*(1-COS(Ψ_1))

VFw1_w2=0.5*(COS(Ψ_1)+COS(Ψ_3))

If $\Psi_{3off}<\Psi_3$, then

VFw1_s=0.5*(1-COS(Ψ_{3off}))

If $\Psi_3$off>Ψ_3, then

VFw1_s=0.5*(1-COS(Ψ_3))

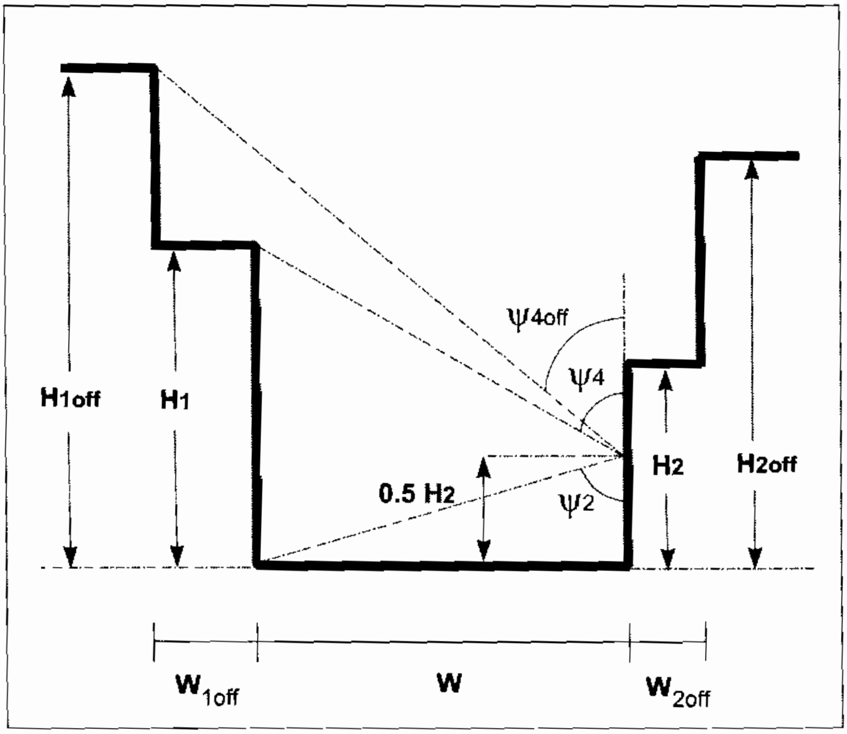

Figure 2: View factors from the middle of canyon wall 2

Ψ_2=ATAN(W/(H_2*0.5))

Ψ_4=ATAN(W/(H_1-0.5*H_1)), but Ψ_4 cannot exceed $\pi/2$.

Ψ_{4off}=ATAN((W+W_{1off})/(H_{1off}-(0.5*H_2)))

VFw2_g=0.5*(1-COS(Ψ_2))

VFw2_w1=0.5*(COS(Ψ_2)+COS(Ψ_4))

If $\Psi_{4off}<\Psi_4$, then

VFw2_s=0.5*(1-COS(Ψ_{4off}))

If $\Psi_4$off>Ψ_4, then

VFw2_s=0.5*(1-COS(Ψ_4))

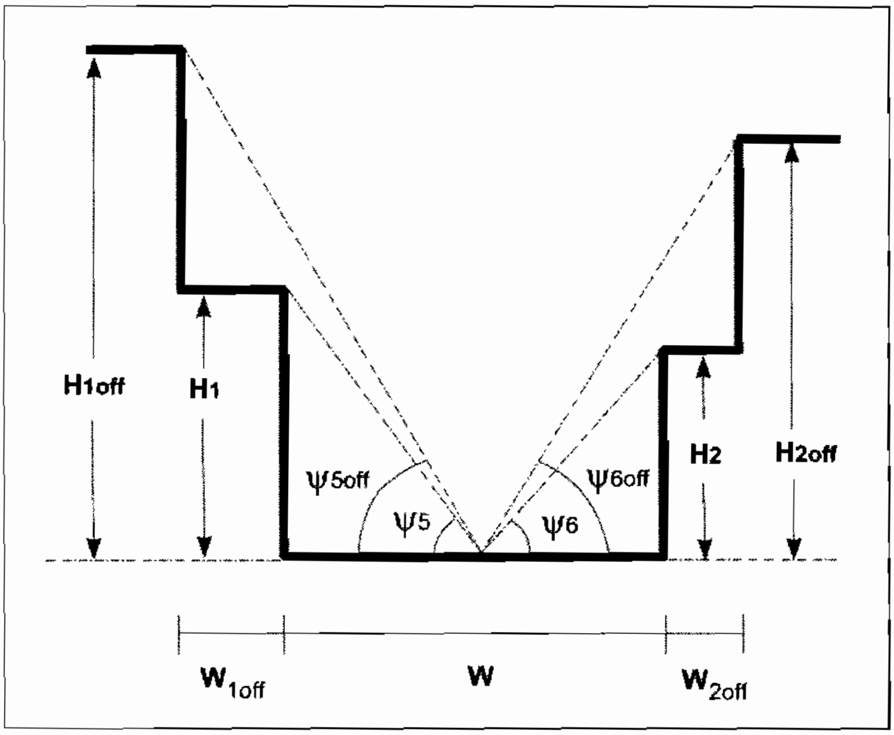

Figure 3: View factors from middle of canyon floor

Ψ_5=ATAN(H_1/(0.5*W))

Ψ_6=ATAN(H_2/(0.5*W))

Ψ_{5off}=ATAN(H_{1off}/(0.5*W+W_{1off}))

Ψ_{6off}=ATAN(H_{2off}/(0.5*W+W_{2off}))

VFg_w1=0.5*(1-COS(Ψ_5))

VFg_w2=0.5*(1-COS(Ψ_6))

To test for restriction of sky view factor by adjacent buildings taller than canyon walls, compare Ψ_5 with Ψ_{5off} and Ψ_6 with Ψ_{6off}.

If, as in Figure 3 above, $\Psi_{5off} > \Psi_5$ and $\Psi_{6off} > \Psi_6$, then

VFg_s=1-[(1-COS(Ψ_{5off})) + (1-COS(Ψ_{6off}))]

APPENDIX B: CALCULATION OF SHADED AREAS

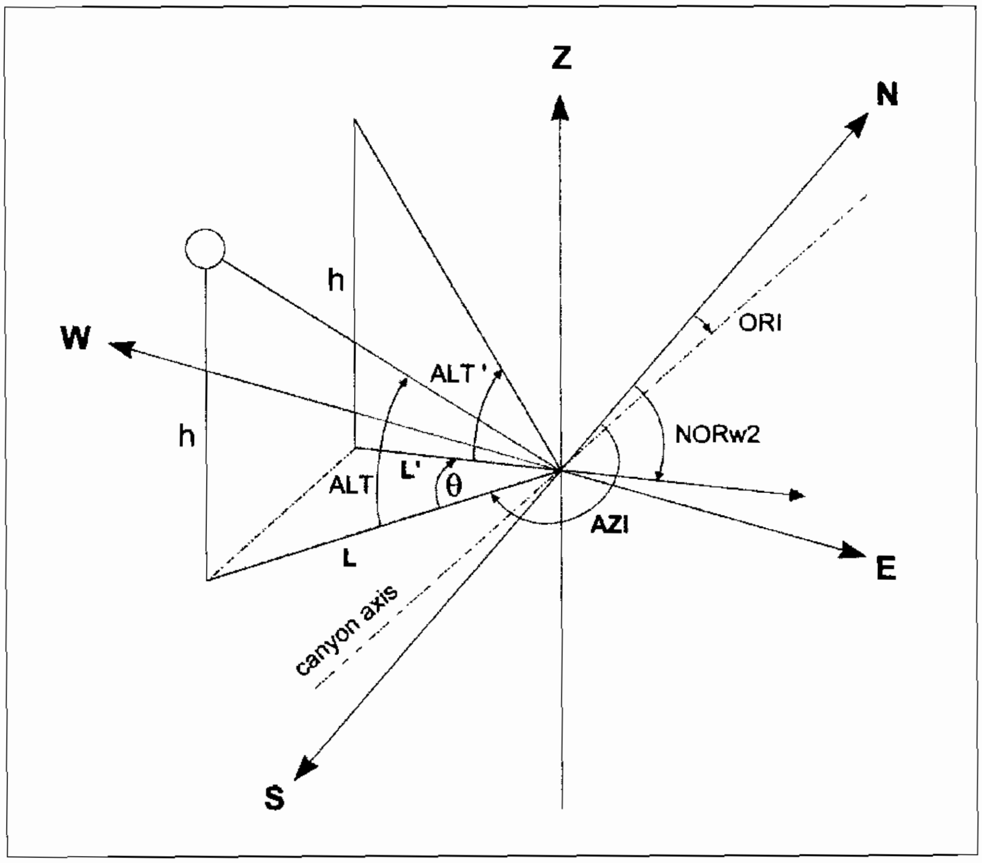

Figure 4: Projection of solar altitude (ALT) on plane perpendicular to canyon axis

$\tan(ALT') = h/L'$

$\theta = |AZI - NORw2|$

$L' = L \cos\theta$

$h = L \tan(ALT)$

$\tan(ALT') = L \tan(ALT) / L \cos\theta$

$\quad\quad = \tan(ALT) / \cos\theta$

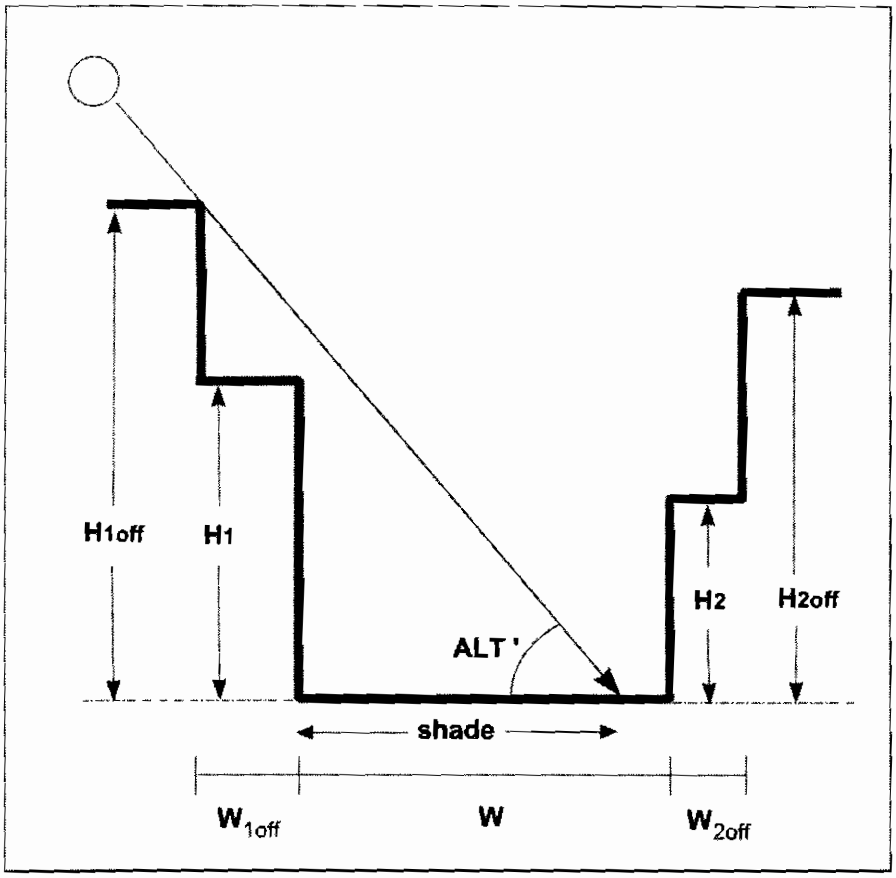

Figure 5: Shaded fraction of canyon floor resulting from obstruction of direct solar radiation by adjacent building

PSAg = shade/W

If the adjacent building casts the shadow,

shade = $(H_{1off} / \tan(ALT')) - W_{1off}$

PSAg = $(([H_{1off} / \tan(ALT')) - W_{1off}) / W$

but

$\tan(ALT') = \tan(ALT) / \cos\theta$

PSAg = $((H_{1off} * \cos\theta / \tan(ALT)) - W_{1off}) / W$

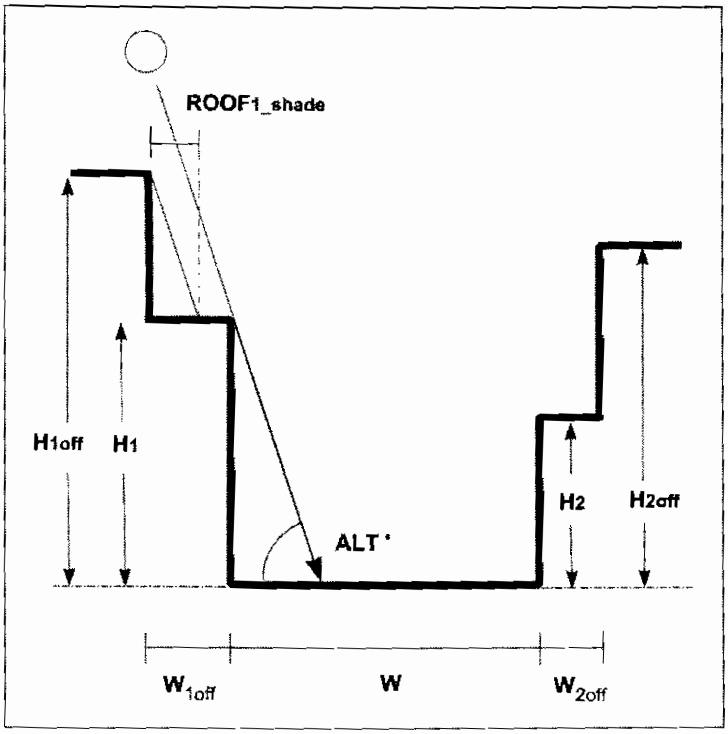

Figure 6: Shaded fraction of canyon floor resulting from obstruction of direct solar radiation by canyon wall

If the canyon wall casts the shadow,

$$shade = H_1 / \tan(ALT')$$

$$PSAg = H_1 / \tan(ALT') / W$$

but

$$\tan(ALT') = \tan(ALT) / \cos\theta$$

$$PSAg = (H_1 / W) * (\cos\theta / \tan(ALT))$$

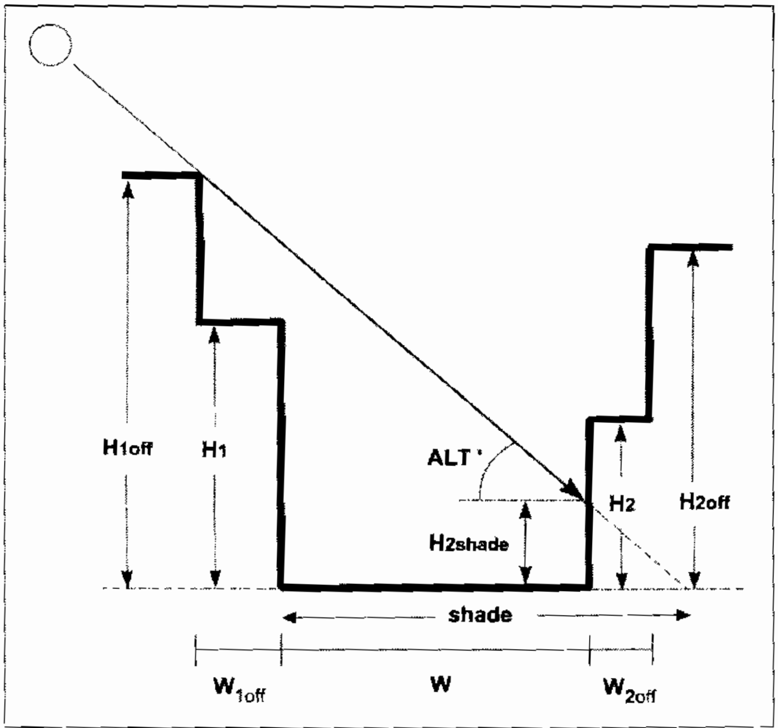

Figure 7: Shaded fraction of canyon wall resulting from obstruction of direct solar radiation by adjacent building

$PSA_{w2} = H_{2shade}/H_2$

If the adjacent tall building casts the shadow,

$H_{2shade} = H_{1off} - \tan(ALT') * (W + W_{1off})$

but

$\tan(ALT') = \tan(ALT) / \cos\theta$

$PSA_{w2} = (1/H_2) * (H_{1off} - (\tan(ALT) / \cos\theta) * (W + W_{1off}))$

If the other canyon wall casts the shadow,

$PSA_{w2} = (1/H_2) * (H_1 - (\tan(ALT) / \cos\theta) * W)$

APPENDIX C: CAT CODE (FORTRAN)

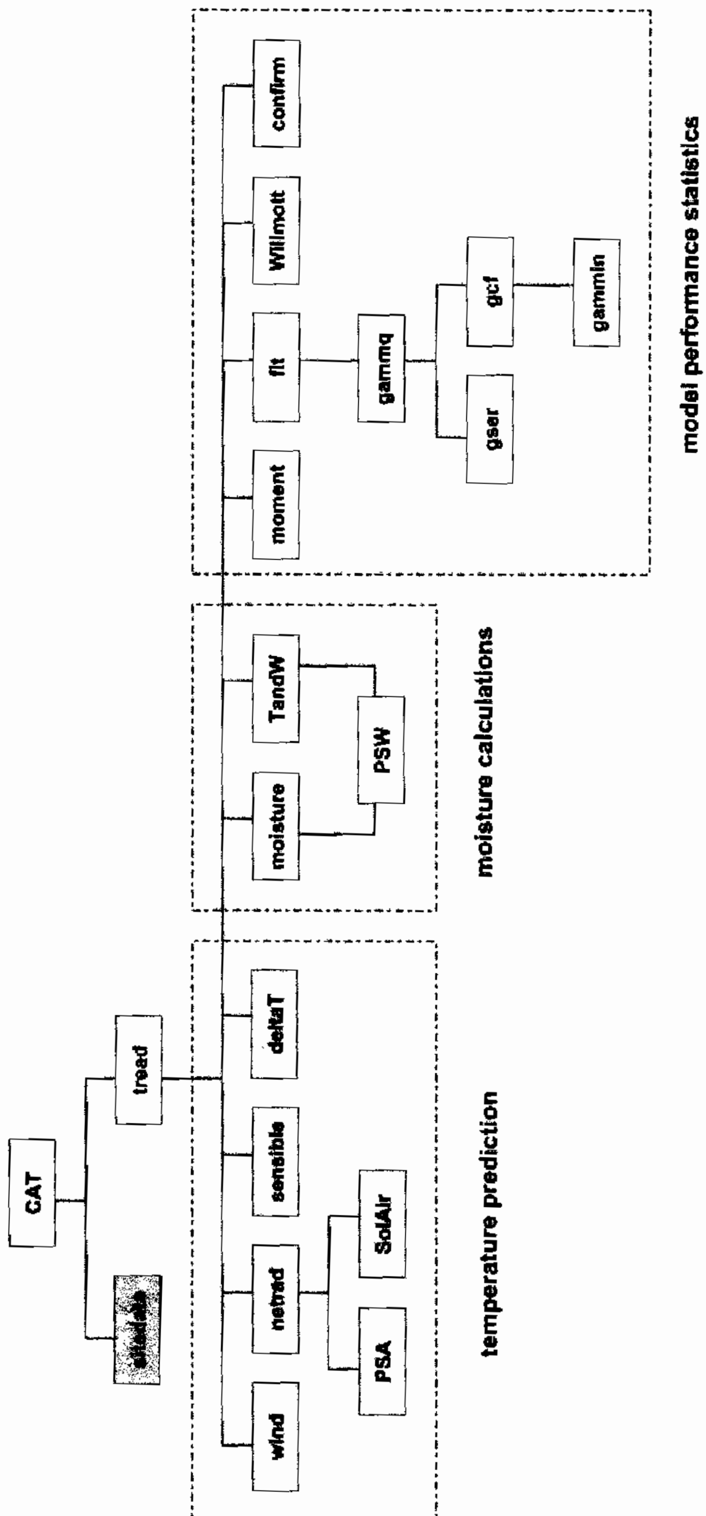

Figure C1: Structure of CAT program

```
PROGRAM CAT

c  Version: 9_14
c  Last change:  EE   September 6, 2004

      REAL albed_met(2),A1_met(3),A2_met(3),A3_met(3)
      REAL albed_urb(2),A1_urb(3),A2_urb(3),A3_urb(3)
      REAL ORIdeg_met,W_met,H1_met,H2_met
      REAL W1off_met,H1off_met,W2off_met,H2off_met
      REAL ORIdeg_urb,W_urb,H1_urb,H2_urb
      REAL W1off_urb,H1off_urb,W2off_urb,H2off_urb
      REAL alpha_met,alpha_urb
      REAL mix_met(4),mix_urb(4)

      INTEGER IMS,IYS,IM,IY

c  Note: The overall structure of this program is based on previous
c  software by Williamson, especially with respect to data input and
c  initial processing. Theoretical basis for computation algorithms
c  of the micro-climate model is acknowledged in the relevant sections.
c  Statistic evaluation subroutines by Williamson.


      print*,'**********************************************************'
      print*,'                                                      '
      print*,'THIS PROGRAM CALCULATES CANYON AIR TEMPERATURE FROM   '
      print*,'CLIMATIC DATA OF REFERENCE SITE                       '
      print*,'                                                      '
      print*,'IMS - START MONTH OF INPUT DATA                       '
      print*,'IYS - START YEAR  OF INPUT DATA                       '
      print*,'IM  - REQUIRED START MONTH FOR SELECTED DATA          '
      print*,'IY  - REQUIRED START YEAR  FOR SELECTED DATA          '
      print*,'                                                      '
      print*,'**********************************************************'

      OPEN(UNIT=1,FILE='meteo_in.txt')
      OPEN(UNIT=2,FILE='predict.txt')
      OPEN(UNIT=4,FILE='met_site.txt')
      OPEN(UNIT=7,FILE='urb_site.txt')
      OPEN(UNIT=8,FILE='netrad.txt')
      OPEN(UNIT=9,FILE='solair.txt')
      OPEN(UNIT=11,FILE='ufluxes.txt')
c     OPEN(UNIT=12,FILE='moisture.txt')
      OPEN(UNIT=13,FILE='stats.txt')
      OPEN(UNIT=14,FILE='averages.txt')
      OPEN(UNIT=15,FILE='wind.txt')
      OPEN(UNIT=16,FILE='mfluxes.txt')

       PRINT 11
   11 FORMAT(/,5x,' input IMS, IYS, IM, IY, NO OF MONTHS - ')
      READ *, IMS,IYS,IM,IY,NMONTH
   12 FORMAT(5I2)

c  read site parameters

      print*
      print*,'Reading site data for the weather data collection site.'

      CALL SITEDATA(4,albed_met,W_met,H1_met,H2_met,
     1 W1off_met,H1off_met,W2off_met,H2off_met,
```

```
     2 ORIdeg_met,A1_met,A2_met,A3_met)

      print*
      print*,'Reading site data for the urban canyon site.'

      CALL SITEDATA(7,albed_urb,W_urb,H1_urb,H2_urb,
     1 W1off_urb,H1off_urb,W2off_urb,H2off_urb,
     2 ORIdeg_urb,A1_urb,A2_urb,A3_urb)

c  Values for the mixing coefficient mix_met and mix_urb are still read
c  from files met_site.txt and urb_site.txt, but the values are not
c  used. Instead, identical values are used for both sites,
c  and are included in subroutine deltaT.


c  Read months prior to required period

      IFLG=1

      MNUM=IM-IMS+(IY-IYS)*12
      IF(MNUM.EQ.0) GOTO 20

       CALL TREAD(IMS,IYS,MNUM,IFLG,
     1 albed_met,W_met,H1_met,H2_met,W1off_met,H1off_met,
     2 W2off_met,H2off_met,ORIdeg_met,
     3 albed_urb,W_urb,H1_urb,H2_urb,W1off_urb,H1off_urb,
     4 W2off_urb,H2off_urb,ORIdeg_urb,
     5 Ta_chesser,dTa_met,dTa_urb,Ta_base,Ta_urb,Qh_met,Qh_urb,
     6 Alpha_met,Alpha_urb,mix_met,mix_urb,
     7 A1_met,A2_met,A3_met,A1_urb,A2_urb,A3_urb)


c  Read data for required period

   20 IFLG=2

      CALL TREAD(IM,IY,NMONTH,IFLG,
     1 albed_met,W_met,H1_met,H2_met,W1off_met,H1off_met,
     2 W2off_met,H2off_met,ORIdeg_met,
     3 albed_urb,W_urb,H1_urb,H2_urb,W1off_urb,H1off_urb,
     4 W2off_urb,H2off_urb,ORIdeg_urb,
     5 Ta_chesser,dTa_met,dTa_urb,Ta_base,Ta_urb,Qh_met,Qh_urb,
     6 Alpha_met,Alpha_urb,mix_met,mix_urb,
     7 A1_met,A2_met,A3_met,A1_urb,A2_urb,A3_urb)


      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CLOSE(UNIT=4)
      CLOSE(UNIT=7)
      CLOSE(UNIT=8)
      CLOSE(UNIT=9)
      CLOSE(UNIT=10)
      CLOSE(UNIT=11)
c     CLOSE(UNIT=12)
      CLOSE(UNIT=13)
      CLOSE(UNIT=14)
      CLOSE(UNIT=15)
      CLOSE(UNIT=16)
```

```
      STOP
      END

C------------------------------------------------------------------

      SUBROUTINE TREAD (IMON,IYEAR,NMON,IFLG,
     1 albed_met,W_met,H1_met,H2_met,W1off_met,H1off_met,
     2 W2off_met,H2off_met,ORIdeg_met,
     3 albed_urb,W_urb,H1_urb,H2_urb,W1off_urb,H1off_urb,
     4 W2off_urb,H2off_urb,ORIdeg_urb,
     5 Ta_chesser,dTa_met,dTa_urb,Ta_base,Ta_urb,Qh_met,Qh_urb,
     6 Alpha_met,Alpha_urb,mix_met,mix_urb,
     7 A1_met,A2_met,A3_met,A1_urb,A2_urb,A3_urb)


c  This subroutine ties together the main program modules and
c  outputs predicted values for canyon air temperature.


      REAL albed_met(2),mix_met(4),MIXING_met,SURFonW_met
      REAL albed_urb(2),mix_urb(4),MIXING_urb,SURFonW_urb
      REAL ORIdeg_met,ORIdeg_urb
      REAL Qstar_gurb,Qstar_w1urb,Qstar_w2urb
      REAL Qstar_gmet,Qstar_w1met,Qstar_w2met
      REAL Ta_met,Ta_urb,Ta_chesser,HeatD(12)
      REAL ALTdeg,AZIdeg,U10_met,RAIN
      REAL Qh_met,Qh_urb,Qe_met,Qe_urb,Ta_error,Qf
      REAL A1_met(3),A2_met(3),A3_met(3),A1_urb(3),A2_urb(3),A3_urb(3)
      REAL moist_met,moist_chesser,moist_base,moist_urb
      REAL Alpha_met,Alpha_urb
      REAL MAvail_met(12),MAvail_urb(12),Rt_met,Rt_urb

      REAL Ta_met_min,Ta_chesser_min,Ta_urb_min
      REAL Ta_met_avg,Ta_chesser_avg,Ta_urb_avg
      REAL Ta_met_max,Ta_chesser_max,Ta_urb_max

      REAL Ta_met_dmin(31),Ta_met_dmax(31)
      REAL Ta_urb_dmin(31),Ta_urb_dmax(31)
      REAL Ta_chesser_dmin(31),Ta_chesser_dmax(31)

      INTEGER yy,mm,dd,hh
      INTEGER RH_met,RH_urb,Udeg_met,CLOUD,Iglob,Idif,RH_chesser
      INTEGER HOURCOUNT

      LOGICAL RAIN_prev

      REAL T_error(9000),T_predict(9000),T_obs(9000),T_input(9000),
     1 SIG(9000)

c  The data input for HeatD is anthropogenic heat, assuming constant
c  monthly value, in W m-2 of canyon floor area.

c     DATA HeatD/15,15,15,15,15,15,15,20,35,35,30,30,30,30,30,35,35,
c    1  35,30,30,25,20,15,15/

      DATA HeatD/15,15,15,20,30,35,35,35,30,25,15,15/


c  The data input for MAvail_met is moisture availability,
```

```
c  assuming constant monthly values.

     DATA MAvail_met/0.25,0.25,0.25,0.6,1.0,1.0,
    1 1.0,0.8,0.6,0.4,0.25,0.25/

c  The data input for MAvail_urb is moisture availability,
c  assuming constant monthly values.

     DATA MAvail_urb/0.25,0.25,0.25,0.25,0.25,0.25,
    1 0.25,0.25,0.25,0.25,0.25,0.25/

c   Initialise variables

     RAIN_prev = .FALSE.
     HOURCOUNT=0
     mtempkount=0.
     Months_Done=0
     Qstar_gurb =0.
     Qstar_w1urb =0.
     Qstar_w2urb =0.
     Qstar_gmet =0.
     Qstar_w1met =0.
     Qstar_w2met =0.
     ERROR_max=0.
     ERROR_min=0.


     T_error(9000) = 0.
     T_predict(9000) = 0.
     T_obs(9000) = 0.
     T_input(9000) =0.
     SIG(9000) =0.


     Ta_met_min=100.
     Ta_chesser_min=100.
     Ta_urb_min=100.

     Ta_met_max=0.
     Ta_chesser_max=0.
     Ta_urb_max=0.

     Ta_met_avg=0.
     Ta_chesser_avg=0.
     Ta_urb_avg=0.

     Ta_met_sum=0.
     Ta_chesser_sum=0.
     Ta_urb_sum=0.

     do i=1,31
      Ta_met_dmin(i)=100.
      Ta_urb_dmin(i)=100.
      Ta_chesser_dmin(i)=100.
      Ta_met_dmax(i)=0.
      Ta_urb_dmax(i)=0.
      Ta_chesser_dmax(i)=0.
     end do

     KStart=1
```

```
 22 DO 7 K=Kstart,NMON
   Months_Done= Months_Done+1


   GOTO(10,20,10,30,10,30,10,10,30,10,30,10) IMON
 10 IDAY=31
   GOTO 50
 20 IDAY=28
   IF(IYEAR.EQ.88) IDAY=29
   IF(IYEAR.EQ.92) IDAY=29
   IF(IYEAR.EQ.96) IDAY=29
   IF(iyear.eq.00) Iday=29
   IF(iyear.eq.04) Iday=29
   GOTO 50
 30 IDAY=30
 50 NF=IDAY*24


 60 DO 3 Icount=1,NF


c  weather data from met station (and Chesset St. DBT and RH)

    READ (1,*,END=90) YY,MM,DD,HH,Ta_met,RH_met,U10_met,Udeg_met,
   1 Cloud,RAIN,Iglob,Idif,ALTdeg,AZIdeg,Ta_chesser,RH_chesser


    IF(HH.eq.0) hh=24

    IF(DD.eq.1.AND.HH.eq.1) then
     print*,'Reading month  ',MM
    ENDIF

    IF(HH.eq.1) then
     numday=(icount/24)+1
     print*,'Reading day ',numday
    ENDIF

    IF(IFLG.EQ.1) GOTO 3

c  If iflg=1 then only stepping through unwanted months
c  so jump to end of subroutine.

c -----------------------------------------------------------------
c    WIND CALCULATIONS FOR SURFACE COEFFICIENTS IN BOTH SITES
c -----------------------------------------------------------------


    CALL WIND (W_urb,H1_urb,H2_urb,U10_met,Udeg_met,ORIdeg_urb,
   1 Hc_gmet,Hc_wmet,Hc_gurb,Hc_wurb,Rt_met,Rt_urb)

c -----------------------------------------------------------------
c    CALCULATIONS FOR REFERENCE SITE:
c -----------------------------------------------------------------

c  Calculate net radiation for reference site:

    CALL NETRAD (4,Ta_met,RH_met,CLOUD,Iglob,Idif,
   1 H1_met,H2_met,W_met,H1off_met,H2off_met,
   2 W1off_met,W2off_met,ORIdeg_met,AZIdeg,ALTdeg,Hc_gmet,HC_wmet,
```

```
     3 Qstar_gmet,Qstar_w1met,Qstar_w2met,
     4 Qstar_gmetprev,Qstar_w1metprev,Qstar_w2metprev,
     5 albed_met(1),albed_met(2),Tsurf_met)


c  Read appropriate monthly value for moisture availability (alpha)

      alpha_met=MAvail_met(mm)


c  Calculate sensible and latent heat flux in reference site:

      CALL SENSIBLE (4,Ta_met,Qstar_gmet,Qstar_gmetprev,
     1 dQstar_gmet,dQs_gmet,Qh_gmet,Qe_gmet,alpha_met,
     2 A1_met(1),A2_met(1),A3_met(1),RAIN,RAIN_prev)

      CALL SENSIBLE (4,Ta_met,Qstar_w1met,Qstar_w1metprev,
     1 dQstar_w1met,dQs_w1met,Qh_w1met,Qe_w1met,alpha_met,
     2 A1_met(2),A2_met(2),A3_met(2),RAIN,RAIN_prev)

      CALL SENSIBLE (4,Ta_met,Qstar_w2met,Qstar_w2metprev,
     1 dQstar_w2met,dQs_w2met,Qh_w2met,Qe_w2met,alpha_met,
     2 A1_met(3),A2_met(3),A3_met(3),RAIN,RAIN_prev)

      Qh_met=(Qh_gmet*W_met+Qh_w1met*H1_met+Qh_w2met*H2_met)/
     1 (W_met)

      Qe_met=(Qe_gmet*W_met+Qe_w1met*H1_met+Qe_w2met*H2_met)/
     1 (W_met)


      SURFonW_met=(W_met+H1_met+H2_met)/W_met
      Vol_met = (W_met*(H1_met+H2_met)/2.)

c  Calculate delta T for the reference site:

      dTsurf=(Tsurf_met-273.15)-Ta_met

      CALL deltaT (Hc_gmet,Qh_met,dTa_met,mix_met,
     1 SURFonW_met,Vol_met,MIXING_met,dTsurf,Qe_met,dMc_met)


c -------------------------------------------------------------------
c    CALCULATIONS FOR URBAN CANYON:
c -------------------------------------------------------------------

c  Calculate net radiation for urban canyon:

      CALL NETRAD (7,Ta_met,RH_met,CLOUD,Iglob,Idif,
     1 H1_urb,H2_urb,W_urb,H1off_urb,H2off_urb,
     2 W1off_urb,W2off_urb,ORIdeg_urb,AZIdeg,ALTdeg,Hc_gurb,Hc_wurb,
     3 Qstar_gurb,Qstar_w1urb,Qstar_w2urb,
     4 Qstar_gurbprev,Qstar_w1urbprev,Qstar_w2urbprev,
     5 albed_urb(1),albed_urb(2),Tsurf_urb)


c  Read appropriate monthly value for moisture availability (alpha)

      alpha_urb=MAvail_urb(mm)
```

```
c Calculate sensible heat flux in urban canyon, beginning with
c individual surfaces:

      CALL SENSIBLE (7,Ta_met,Qstar_gurb,Qstar_gurbprev,
     1 dQstar_gurb,dQs_gurb,Qh_gurb,Qe_gurb,alpha_urb,
     2 A1_urb(1),A2_urb(1),A3_urb(1),RAIN,RAIN_prev)

      CALL SENSIBLE (7,Ta_met,Qstar_w1urb,Qstar_w1urbprev,
     1 dQstar_w1urb,dQs_w1urb,Qh_w1urb,Qe_w1urb,alpha_urb,
     2 A1_urb(2),A2_urb(2),A3_urb(2),RAIN,RAIN_prev)

      CALL SENSIBLE (7,Ta_met,Qstar_w2urb,Qstar_w2urbprev,
     1 dQstar_w2urb,dQs_w2urb,Qh_w2urb,Qe_w2urb,alpha_urb,
     2 A1_urb(3),A2_urb(3),A3_urb(3),RAIN,RAIN_prev)


c      Qf=HeatD(mm)

c The following correction for the diurnal pattern of anthropogenic
c heat is an approximation based on Sailor and Lu's (2004) paper in
c Atmospheric Environment.

      IF(HH.GT.8 .AND. HH.LT.17) THEN
       Qf=HeatD(mm)
      ELSE
       Qf=HeatD(mm)*0.5
      ENDIF

c Following is a test for rain in the previous hour, incorporated
c in increase value of moisture availability for current hour.

      IF(RAIN.GT.0.5) THEN
       RAIN_prev = .TRUE.
      ELSE
       RAIN_prev = .FALSE.
      END IF

c To add anthropogenic heat to the calculation, enable following
c expression for Qh, which includes Qf.

      Qh_urb=Qf+((Qh_gurb*W_urb+Qh_w1urb*H1_urb+Qh_w2urb*H2_urb)/
     1 (W_urb))

      Qe_urb=((Qe_gurb*W_urb+Qe_w1urb*H1_urb+Qe_w2urb*H2_urb)/
     1 (W_urb))


c Write fluxes to diagnostic files UFLUXES.TXT and MFLUXES.MET:

      WRITE(11,559) mm,dd,hh,Qh_urb,Qstar_gurb,dQs_gurb,Qh_gurb,
     1 Qstar_w1urb,dQs_w1urb,Qh_w1urb,Qstar_w2urb,dQs_w2urb,Qh_w2urb,
     2 Qf


      WRITE(16,559) mm,dd,hh,Qh_met,Qstar_gmet,dQs_gmet,Qh_gmet,
     1 Qstar_w1met,dQs_w1met,Qh_w1met,Qstar_w2met,dQs_w2met,Qh_w2met


 559  FORMAT(3(I5),11(f7.2))
```

```
c  Calculate delta T for the urban canyon:

      SURFonW_urb=(W_urb+H1_urb+H2_urb)/W_urb
      Vol_urb=(W_urb*(H1_urb+H2_urb)/2.)

      CALL deltaT (Hc_gurb,Qh_urb,dTa_urb,mix_urb,
     1 SURFonW_urb,Vol_urb,MIXING_urb,dTsurf,Qe_urb,dMc_urb)

c  Write wind data and calculated factors to diagnostic file WIND.TXT:

      WRITE(15,589) mm,dd,hh,U10_met,Hc_gmet,Hc_wmet,
     1 Hc_gurb,Hc_wurb,mixing_met

 589  FORMAT(3(I4),F5.1,4(f7.2),f5.2)

c ----------------------------------------------------------------
c    CALCULATIONS FOR CANYON AIR TEMPERATURE:
c ----------------------------------------------------------------

c  Calculate base temperature and then canyon air temperature.
c  Concept from Elnahas and Williamson, 1997:

      Ta_base=Ta_met-dTa_met
      Ta_urb=Ta_base+dTa_urb

c  Assemble data for error statistics

      Ta_error = Ta_chesser-Ta_urb

      IF(Ta_error.GT.ERROR_max) ERROR_max=Ta_error
      IF(Ta_error.LT.ERROR_min) ERROR_min=Ta_error

      HOURCOUNT=HOURCOUNT+1

      T_error(HOURCOUNT)=Ta_error
      T_predict(HOURCOUNT) = Ta_urb
      T_Obs(HOURCOUNT) = Ta_Chesser
      T_input(HOURCOUNT) = Ta_met

      IF (Ta_met.LT.Ta_met_min) Ta_met_min=Ta_met
      IF (Ta_met.GT.Ta_met_max) Ta_met_max=Ta_met

      IF (Ta_chesser.LT.Ta_chesser_min) Ta_chesser_min=Ta_chesser
      IF (Ta_chesser.GT.Ta_chesser_max) Ta_chesser_max=Ta_chesser

      IF (Ta_urb.LT.Ta_urb_min) Ta_urb_min=Ta_urb
      IF (Ta_urb.GT.Ta_urb_max) Ta_urb_max=Ta_urb

      Ta_met_sum=Ta_met_sum+Ta_met
      Ta_met_avg=Ta_met_sum/HOURCOUNT

      Ta_chesser_sum=Ta_chesser_sum+Ta_chesser
      Ta_chesser_avg=Ta_chesser_sum/HOURCOUNT
```

```
      Ta_urb_sum=Ta_urb_sum+Ta_urb
      Ta_urb_avg=Ta_urb_sum/HOURCOUNT


      IF (Ta_met.LT.Ta_met_dmin(dd)) Ta_met_dmin(dd)=Ta_met
      IF (Ta_urb.LT.Ta_urb_dmin(dd)) Ta_urb_dmin(dd)=Ta_met
      IF (Ta_chesser.LT.Ta_chesser_dmin(dd)) THEN
       Ta_chesser_dmin(dd)=Ta_met
      END IF
      IF (Ta_met.GT.Ta_met_dmax(dd)) Ta_met_dmax(dd)=Ta_met
      IF (Ta_urb.GT.Ta_urb_dmax(dd)) Ta_urb_dmax(dd)=Ta_urb
      IF (Ta_chesser.GT.Ta_chesser_dmax(dd)) THEN
       Ta_chesser_dmax(dd)=Ta_chesser
      END IF

c -----------------------------------------------------------------
c   CALCULATIONS FOR MOISTURE CONTENT:
c -----------------------------------------------------------------


c First calculate moisture content from DBT and RH.
c Data for Chesser site used for program validation only.


      CALL MOISTURE (1013.25,Ta_met,RH_met,moist_met)

      CALL MOISTURE (1013.25,Ta_chesser,RH_chesser,moist_chesser)

      moist_base=moist_met-dMc_met
      moist_urb=moist_base+dMc_urb

      CALL TandW (Ta_urb,moist_urb,RH_urb)


c     PRINT*,"Ta_met,RH_met,Ta_urb,RH_urb"
c     print*,Ta_met,RH_met,Ta_urb,RH_urb
c     PRINT*,"dMc_met,dMc_urb,moist_met,moist_chesser,moist_urb"
c     print*, dMc_met,dMc_urb,moist_met,moist_chesser,moist_urb
c     pause

c Write moisture to diagnostic file MOISTURE.TXT

      WRITE(12,560) mm,dd,hh,Ta_met,RH_met,moist_met,
     1 Ta_chesser,RH_chesser,moist_chesser,Ta_urb,RH_urb,moist_urb

 560  FORMAT(1x,3I5,3(f7.2,I5,f8.4))


c Write data to output file PREDICT.TXT:

      WRITE(2,120) yy,mm,dd,hh,Ta_met,U10_met,Udeg_met,Cloud,
     1 Iglob,Idif,Ta_chesser,Ta_base,Ta_urb,Ta_error,
     2 dTa_met,dTa_urb

  120 FORMAT(4(I3),2(F5.1),I4,I2,2(I5),3(F5.1),F6.1,2(F5.1))


      mtempkount=mtempkount+1
```

```
  3 CONTINUE

    imon=imon+1
    IF(imon.GT.12) THEN
     imon=1
     iyear=iyear+1
    ENDIF

  7 CONTINUE

    IF(IFLG.EQ.1) GOTO 99

c check if all months are done, rewind or exit

 90 IF (Months_Done.ne.NMON) THEN
     REWIND 1
     imon=imon+1
     IF(imon.GT.12) THEN
     imon=1
    ENDIF

    Kstart=Months_Done+1
    GOTO 22
    ELSE
    CONTINUE
    ENDIF


    DO dd=1,31
     Ta_met_dmin_sum=Ta_met_dmin_sum+Ta_met_dmin(dd)
     Ta_urb_dmin_sum=Ta_urb_dmin_sum+Ta_urb_dmin(dd)
     Ta_chesser_dmin_sum=Ta_chesser_dmin_sum+Ta_chesser_dmin(dd)
     Ta_met_dmax_sum=Ta_met_dmax_sum+Ta_met_dmax(dd)
     Ta_urb_dmax_sum=Ta_urb_dmax_sum+Ta_urb_dmax(dd)
     Ta_chesser_dmax_sum=Ta_chesser_dmax_sum+Ta_chesser_dmax(dd)
    END DO

    Ta_met_dmin_avg=Ta_met_dmin_sum/Iday
    Ta_urb_dmin_avg=Ta_urb_dmin_sum/Iday
    Ta_chesser_dmin_avg=Ta_chesser_dmin_sum/Iday
    Ta_met_dmax_avg=Ta_met_dmax_sum/Iday
    Ta_urb_dmax_avg=Ta_urb_dmax_sum/Iday
    Ta_chesser_dmax_avg=Ta_chesser_dmax_sum/Iday


    CALL MOMENT(T_error,HOURCOUNT,AVE,ADEV,SDEV,VAR,SKEW,CURT)

c
c  Calculate Willmott statistics
c
c  initial SIG to zero values
c  set mwt =0
c
    do i=1,9000
    sig(i) =0.
    end do
c
    mwt=0.

    CALL FIT(T_obs,T_predict,HOURCOUNT,SIG,mwt,A,B,SIGA,SIGB,CHI2,Q)
```

```
c calcluate Willmott systematic and unsystematic errors (RMSEs & RMSEu)
c and index of agreement (d)
c
      call Willmott(T_predict,T_obs,HOURCOUNT,A,B,RMSE,RMSEs,RMSEu,d,
     1 wMSE,wMSEs,wMSEu)

      call Confirm(T_predict,T_obs,T_input,HOURCOUNT,Theil,DofC)
c       PRINT*,"U= ",U,"D= ",DofC


      PRINT*,
      PRINT*,'**************************************************'
      PRINT*,'     ERROR STATISTICS'
      PRINT*,'**************************************************'
      PRINT*,
      PRINT 100, HOURCOUNT
  100 FORMAT ( 5x,'total number of hours      = ',I5)
      PRINT 101,AVE
  101 FORMAT ( 5x,'mean error                 = ',F5.2)
      PRINT 102,SDEV
  102 FORMAT ( 5x,'standard deviation         = ',F5.2)
      PRINT 103,ERROR_max
  103 FORMAT ( 5x,'maximum error              = ',F5.2)
      PRINT 104,ERROR_min
  104 FORMAT ( 5x,'minimum error              = ',F5.2)
      PRINT 105,wMSE
  105 FORMAT ( 5x,'MSE                        = ',F6.3)
      PRINT 106,wMSEs
  106 FORMAT ( 5x,'systematic MSE             = ',F6.3)
      PRINT 107,wMSEu
  107 FORMAT ( 5x,'unsystematic MSE           = ',F6.3)
      PRINT 108,d
  108 FORMAT ( 5x,'Wilmott index              = ',F6.3)
      PRINT 109,Theil
  109 FORMAT ( 5x,'Theil inequality coeff.    = ',F6.3)
      PRINT 110,DofC
  110 FORMAT ( 5x,'degree of confirmation     = ',F6.3)
      PRINT*,

      PRINT*,'Data below only valid for simulation of single months.'
      PRINT*,
      PRINT*,'                              City       City'
      PRINT*,'                      BoM      obs.      pred.'
      PRINT 111,Ta_met_min,Ta_chesser_min,Ta_urb_min
  111 FORMAT (5x,'abs. monthly min',4x,F5.2,6x,F5.2,6x,F5.2)
      PRINT 112,Ta_met_dmin_avg,Ta_chesser_dmin_avg,
     1 Ta_urb_dmin_avg
  112 FORMAT (5x,'mean monthly min',4x,F5.2,6x,F5.2,6x,F5.2)
      PRINT 113,Ta_met_avg,Ta_chesser_avg,Ta_urb_avg
  113 FORMAT (5x,'monthly avg',9x,F5.2,6x,F5.2,6x,F5.2)
      PRINT 114,Ta_met_dmax_avg,Ta_chesser_dmax_avg,
     1 Ta_urb_dmax_avg
  114 FORMAT (5x,'mean monthly max',4x,F5.2,6x,F5.2,6x,F5.2)
      PRINT 115,Ta_met_max,Ta_chesser_max,Ta_urb_max
  115 FORMAT (5x,'abs. monthly max',4x,F5.2,6x,F5.2,6x,F5.2)
      PRINT*,
      PRINT*,'**************************************************'

      WRITE (13,100) HOURCOUNT
```

```
      WRITE (13,101) AVE
      WRITE (13,102) SDEV
      WRITE (13,103) ERROR_max
      WRITE (13,104) ERROR_min
      WRITE (13,105) wMSE
      WRITE (13,106) wMSEs
      WRITE (13,107) wMSEu
      WRITE (13,108) d
      WRITE (13,109) Theil
      WRITE (13,110) DofC
      WRITE (14,111) Ta_met_min,Ta_chesser_min,Ta_urb_min
      WRITE (14,112) Ta_met_dmin_avg,Ta_chesser_dmin_avg,
     1 Ta_urb_dmin_avg
      WRITE (14,113) Ta_met_avg,Ta_chesser_avg,Ta_urb_avg
      WRITE (14,114) Ta_met_dmax_avg,Ta_chesser_dmax_avg,
     1 Ta_urb_dmax_avg
      WRITE (14,115) Ta_met_max,Ta_chesser_max,Ta_urb_max

   99 RETURN
      END

c-------------------------------------------------------------------

      SUBROUTINE NETRAD (SITE,Ta,RH,CLOUD,Iglob,Idif,
     1 H1,H2,W,H1off,H2off,
     2 W1off,W2off,ORIdeg,AZIdeg,ALTdeg,Hc_g,Hc_w,
     3 Qstarg,Qstarw1,Qstarw2,Qstarg_prev,Qstarw1_prev,Qstarw2_prev,
     4 ALBEDOg,ALBEDOw1,Tsol_g)


c  This subroutine calculates net radiation on surfaces of canyon from
c  measured air DBT, RH, wind speed and wind direction, cloud cover,
c  and global and diffuse solar radiation.

      REAL Ta,TaK,Pw,Pws,ORI,AZI,ALT,
     1 Ibeam,CL10,
     2 Hc_g, Hc_w,Hcr_g,Hcr_w,
     3 Esky,Esurf,Lclear,Lsky,
     4 NORw1,NORw2,ALBEDOg,ALBEDOw1,ALBEDOw2,
     5 VFg_s,VFg_w1,VFg_w2,VFw1_s,VFw1_g,VFw1_w2,VFw2_s,VFw2_g,VFw2_w1,
     6 Kg,Kdirg,Krefg,Kdifg,Kw1,Kdirw1,Kdifw1,
     7 Krefw1,Kw2,Kdirw2,Kdifw2,Krefw2,
     8 Qstarg,Qstarw1,Qstarw2,
     9 H1,H2,W,H1off,H2off,W1off,W2off,AZIdeg,ALTdeg,ORIdeg,Tsol_g

      DOUBLE PRECISION C8,C9,C10,C11,C12,C13,logPws,TaK_D

      INTEGER SITE,CLOUD,RH,Iglob,Idif


c  TIME VARIABLE (ENVIRONMENTAL) INPUTS:
c  Ta is air DBT (deg C)
c  RH is relative humidity (%)
c  U10_met is wind speed at ten metre height (m s-1)
c  Udeg is wind direction (degrees)
c  CLOUD is cloud cover (tenths)
c  Iglob is global radiation on a horizontal surface (W m-2)
c  Ibeam is beam (normal) radiation (W m-2)
c  Idif is diffuse radiation on a horizontal surface (W m-2)
c  AZIdeg is solar azimuth (degrees)
```

```
c  ALTdeg is solar elevation (degrees)

c  CONSTANT INPUTS:
c  W is canyon width (m)
c  H1 is height of canyon wall 1 (m)
c  H2 is height of canyon wall 2 (m)
c  H1off is height of building adjacent to wall 1 (m)
c  H2off is height of building adjacent to wall 2 (m)
c  W1off is offset of building adjacent to wall 1 (m)
c  W2off is offset of building adjacent to wall 2 (m)
c  ALBEDOg is albedo of ground
c  ALBEDOw1 is albedo of wall 1
c  ALBEDOw2 is albedo of wall 2
c  Esurf is long wave emissivity of surface
c  ORIdeg is canyon orientation (degrees)
c  Zd is zero plane displacement (m)
c  Z0 is roughness length (m)
c  Zr is height of anemometer (m)

c  CONSTANTS CALCULATED BY PROGRAM:
c  ORI is canyon orientation (radians)
c  NORw1 is normal to wall 1 (radians)
c  NORw2 is normal to wall 2 (radians)
c  VFg_s is view factor from ground to sky
c  VFg_w1 is view factor from ground to wall 1
c  VFg_w2 is view factor from ground to wall 2
c  VFw1_s is view factor from wall 1 to sky
c  VFw1_g is view factor from wall 1 to ground
c  VFw1_w2 is view factor from wall 1 to wall 2
c  VFw2_s is view factor from wall 2 to sky
c  VFw2_g is view factor from wall 2 to ground
c  VFw2_w1 is view factor from wall 2 to wall 1


c  VARIABLES CALCULATED BY PROGRAM:
c  AZI is solar azimuth (radians)
c  ALT is solar elevation (radians)
c  TaK is air DBT (K)
c  Tsol is sol-air temperature (K)
c  Tsurf is surface temperature calculated from sol-air (K)
c  Pw is partial presure of water vapour (Pa)
c  Pws is partial pressure of water vapour at saturation (Pa)
c  Ustar is friction velocity (m s-1)
c  Hc is surface convective heat exchange coef. (W m-2 K-1)
c  Esky is sky emissivity
c  Lclear is clear sky long wave radiation (W m-2)
c  Lsky is cloudy sky long wave radiation (W m-2)
c  Lstar is net long wave radiation at the surface (W m-2)
c  Lsurf is long wave radiation emitted by surface (W m-2)
c  Kdown is solar radiation on surface (W m-2)
c  Kstar is net shortwave radiation at the surface (W m-2)
c  Kdirg is direct solar radiation on canyon floor (W m-2)
c  Krefg is reflected solar radiation on canyon floor (W m-2)
c  Kdifg is diffuse solar radiation on canyon floor (W m-2)
c  Kg is total solar radiation on canyon floor (W m-2)
c  Kdirw1 is direct solar radiation on wall1 (W m-2)
c  Krefw1 is reflected solar radiation on wall 1 (W m-2)
c  Kdifw1 is diffuse solar radiation on wall 1 (W m-2)
c  Kw1 is total solar radiation on wall 1 (Wm-2)
c  Kdirw2 is direct solar radiation on wall 2 (W m-2)
```

```
c  Krefw2 is reflected solar radiation on wall 2 (W m-2)
c  Kdifw2 is diffuse solar radiation on wall 2 (W m-2)
c  Kw2 is total solar radiation on wall 2 (W m-2)
c  PSAg is partial shaded area ground
c  PSAw1 is partial shaded area of wall 1
c  PSAw2 is partial shaded area of wall 2
c  Qstarg, Qstarw1 and Qstarw2 are net radiation at ground and walls (W m-2)


c  Universal constants:

      PI=3.1415926
      SIGMA = 5.67E-08

c  At this stage, assign equal albedo for both walls:

      ALBEDOw2=ALBEDOw1


      IF (ORIdeg.LE.90) THEN
        ORI=ORIdeg*0.017453
      ELSE
        ORI=(360-ORIdeg)*(-0.017453)
      ENDIF

c  Normal to walls required for solar radiation on wall surfaces:

      IF (ORI.LT.0) THEN
        NORw1=ORI+(PI/2)
        NORw2=ORI-(PI/2)
      ELSE
        NORw1=ORI-(PI/2)
        NORw2=ORI+(PI/2)
      ENDIF

c  Calculate view factors using six angles:

c  PSI1 is view angle from middle of wall 1 to canyon floor
c  PSI2 is view angle from middle of wall 2 to canyon floor
c  PSI3 is view angle from middle of wall 1 to sky
c  PSI4 is view angle from middle of wall 2 to sky
c  PSI5 is view angle from middle of canyon floor to wall 1
c  PSI6 is view angle from middle of canyon floor to wall 2


      PSI1=ATAN(W/(H1*0.5))
      PSI2=ATAN(W/(H2*0.5))
      PSI3=ATAN(W/(H2-0.5*H1))
      PSI4=ATAN(W/(H1-0.5*H2))
      PSI5=ATAN(H1/(0.5*W))
      PSI6=ATAN(H2/(0.5*W))


c  Calculate view factors:

      VFg_w1=0.5*(1-COS(PSI5))
      VFg_w2=0.5*(1-COS(PSI6))

      VFw1_g=0.5*(1-COS(PSI1))
      VFw1_w2=0.5*(COS(PSI1)+COS(PSI3))
```

```
      VFw2_g=0.5*(1-COS(PSI2))
      VFw2_w1=0.5*(COS(PSI2)+COS(PSI4))

c *****************Diagnostic*******************************************
c     Print*,'v.factors',VFg_w1,VFg_w2,VFw1_g,VFw1_w2,VFw2_g,VFw2_w1
c     PAUSE
c ********************************************************************

c  Now test for restriction of view factor to sky by adjacent buildings:

      PSI3off=ATAN((W+W2off)/(H2off-(0.5*H1)))
      IF (PSI3off.LT.PSI3) PSI3=PSI3off

      PSI4off=ATAN((W+W1off)/(H1off-(0.5*H2)))
      IF (PSI4off.LT.PSI4) PSI4=PSI4off

      PSI5off=ATAN(H1off/(0.5*W+W1off))
      IF (PSI5off.GT.PSI5) PSI5=PSI5off

      PSI6off=ATAN(H2off/(0.5*W+W2off))
      IF (PSI6off.GT.PSI6) PSI6=PSI6off


      VFg_s=1-(1-(COS(PSI5)+COS(PSI6))/2)
      VFw1_s=0.5*(1-COS(PSI3))
      VFw2_s=0.5*(1-COS(PSI4))


c  -------------------------------------------------------------------
c  First calculate long wave radiation from sky:
c  -------------------------------------------------------------------

c  constants C8 through C13 are used to calculate vapour pressure (Pw).

      C8  = -5.8002206E+03
      C9  = 1.391449
      C10 = -4.864023E-02
      C11 = 4.176476E-05
      C12 = -1.445209E-08
      C13 = 6.545967

c  calculate partial pressure of water vapour, Pw (Pa),
c  from ASHRAE Fundamentals Chapter 6

      TaK=Ta+273.15
      TaK_D= DBLE(TaK)
      logPws =C8/TaK_D+C9+C10*TaK_D+C11*TaK_D**2+C12*TaK_D**3+
     1 C13*DLOG(TaK_D)
      Pws = SNGL(EXP(logPws))
      Pw = Pws*RH/100

c  To convert Pw in Pa to hPa (mb), divide by 100
      Pw_mb = Pw/100.

c  Atmospheric emissivity calculated from Brutsaert, 1982:

      Esky = 1.24*(Pw_mb/TaK)**(1./7.)

      Lclear = SIGMA*Esky*TaK**4
```

```
c  Cloud correction from Martin, in Cook (ed) Passive Cooling, 1989:
c  Note CLOUD cover in tenths
     Cl10 = FLOAT(CLOUD)

     Lsky = Lclear*(1+0.0224*CL10-0.0035*CL10**2+0.00028*CL10**3)

c      CALL WIND (SITE,W,H1,H2,U10,Udeg,Hcr_g,Hcr_w,Hc,ORIdeg,TaK)

c  -----------------------------------------------------------------
c  Calculate incoming solar radiation on each canyon surface:
c  -----------------------------------------------------------------

c  First convert apparent solar position from degrees to radians.
c  (Solar azimuth is represented as radians east or west of north.)


     IF (AZIdeg.LT.180) THEN
       AZI=AZIdeg*0.017453
     ELSE
       AZI=-(2.*PI-AZIdeg*0.017453)
     ENDIF


     ALT=ALTdeg*0.017453

     IF (ALT.lt.1E-04) THEN
       ALT=0.01
     ENDIF

     Ibeam=(FLOAT(Iglob)-FLOAT(Idif))/SIN(ALT)

     IF(Ibeam.LT.0) THEN
      Ibeam=0
     ENDIF

c  Calculate proportion of each surface shaded
c  from direct solar radiation:

     CALL PSA (H1,H2,W,H1off,H2off,
    1 W1off,W2off,ALT,AZI,PI,
    2 Iglob,NORw1,PSAg,PSAw1,PSAw2)


c  direct radiation on canyon surfaces

     Kdirg=Ibeam*(1-PSAg)*SIN(ALT)

     IF (ABS(AZI-NORw1).LT.(PI/2)) THEN
       Kdirw1=Ibeam*(1-PSAw1)*COS(ALT)*SIN(ABS(AZI-NORw1))
     ELSE
       Kdirw1=0
     ENDIF

     IF (ABS(AZI-NORw2).LT.(PI/2)) THEN
       Kdirw2=Ibeam*(1-PSAw2)*COS(ALT)*SIN(ABS(AZI-NORw2))
     ELSE
       Kdirw2=0
     END IF
```

```
c  diffuse radiation on canyon surfaces

      Kdifg=float(Idif)*VFg_s
      Kdifw1=float(Idif)*VFw1_s
      Kdifw2=float(Idif)*VFw2_s

c  reflected radiation on canyon surfaces

      Krefg=(Kdirw1+Kdifw1)*ALBEDOw1*VFg_w1
     1 +(Kdirw2+Kdifw2)*ALBEDOw2*VFg_w2

      Krefw1=(Kdirg+Kdifg)*ALBEDOg*VFw1_g
     1 +(Kdirw2+Kdifw2)*ALBEDOw2*VFw1_w2

      Krefw2=(Kdirg+Kdifg)*ALBEDOg*VFw2_g
     1 +(Kdirw1+Kdifw1)*ALBEDOw1*VFw2_w1

c  total incoming solar radiation on canyon surfaces

      Kg=(Kdirg+KdifG+Krefg)
      Kw1=(Kdirw1+Kdifw1+Krefw1)
      Kw2=(Kdirw2+Kdifw2+Krefw2)


c *****************Diagnostic******************************************
c     Print*, Ibeam,Idif,Kg,Kw1,Kw2
c     pause
c *********************************************************************


c  Now begin loop for numeric solution of approx. surface temperature
c  using sol-air temperature formula.
c  Use uniform long wave emissivity for all surfaces:

      Esurf=0.95

      Qstarg_prev=Qstarg
      Qstarw1_prev=Qstarw1
      Qstarw2_prev=Qstarw2

c  Add radiative component to convective coeff. for total surface coeff.

      Hr = 4.*0.9*sigma*TaK**3.

      Hcr_g=Hc_g+Hr
      Hcr_w=Hc_w+Hr


      CALL SOLAIR(SITE,Lsky,Kg,TaK,Qstarg,Hcr_g,Esurf,ALBEDOg,VFg_s,
     1 Tsol_g)


      CALL SOLAIR(SITE,Lsky,Kw1,TaK,Qstarw1,Hcr_w,Esurf,ALBEDOw1,VFw1_s,
     1 Tsol_w1)


      CALL SOLAIR(SITE,Lsky,Kw2,TaK,Qstarw2,Hcr_w,Esurf,ALBEDOw2,VFw2_s,
     1 Tsol_w2)
```

```
c *****************Diagnostic******************************************
c    Print*, Qstarg,Qstarw1,Qstarw2
c    pause
c ********************************************************************


c Write solar radiation on canyon surfaces to diagnostic file
c NETRAD.TXT:

     WRITE(8,800) SITE,Iglob,Idif,Kdirg,Krefg,Kdifg,Kg,
    1 Kdirw1,Krefw1,Kdifw1,Kw1,Kdirw2,Krefw2,Kdifw2,Kw2,
    2 Qstarg,Qstarw1,Qstarw2,Tsol_g,Tsol_w1,Tsol_w2

  800 FORMAT(I3,2(I4),18(F7.1))


     RETURN
     END


**********************************************************************


     SUBROUTINE SOLAIR (SITE,Lsky_d,K,TaK_d,Qstar,
    1 Hconrad,Esurf_d,ALBEDO,VF,Tsurf)

     REAL Lsky_d,Lsurf,Lstar,Kstar,K,TaK_d,Tsurf,Tsol,
    1 Qstar,Hconrad,ALBEDO,SIGMA,VF

     INTEGER SITE


c This subroutine calculates the sol-air temperature of a surface.
c The sol-air temperature is used as a surrogate for the actual
c temperature of the surface.


     SIGMA=5.67E-08

     Lsurf=SIGMA*Esurf_d*Tak_d**4


c The following statement assumes no NET exchange
c with other canyon surfaces:

     Lstar=(Lsky_d-Lsurf)*VF
     Kstar=K*(1-ALBEDO)

     Tsurf=TaK_d
     Tsol=TaK_d+(Kstar+Lstar)/Hconrad


     KOUNT=0

   99 ERROR=ABS(Tsol-Tsurf)

     IF (KOUNT.eq.50) goto 98


     IF (ERROR.GT.1.0) THEN
```

```
      KOUNT=KOUNT+1
      Tsurf=Tsurf+0.5*(Tsol-Tsurf)
      Lsurf=Sigma*Esurf_d*Tsurf**4
      Lstar=(Lsky_d-Lsurf)*VF
      Tsol=TaK_d+(Kstar+Lstar)/Hconrad
      GO TO 99
    END IF
    GO TO 97

 98 PRINT*, 'solair did not converge'
    STOP

 97 Qstar=Kstar+Lstar


c  Write interim radiation values used in calculation of sol-air
c  temperature to diagnostic file SOLAIR.TXT:

    write (9,401) SITE,K,Kstar,Lstar,Lsurf,Tsol
401 FORMAT(I2,5(F10.3))
    RETURN

    END
************************************************************************


    SUBROUTINE WIND (W_urb,H1_urb,H2_urb,WIND10,WINDdeg,ORIdeg,
   1 Hc_gmet,Hc_wmet,Hc_gurb,Hc_wurb,Rtop_met,Rtop_urb)

c  This subroutine calculates wind speed near the surfaces of an urban
c  canyon. It is based on the Harlow & Hotchkiss model, and requires
c  measured wind speed above the canyon, wind direction and canyon
c  geometry.


    REAL  WIND10,Uroof,Ustar,Uh,Vh,
   1 W_urb,H1_urb,H2_urb,Uxz,Vz,Wxz,Ures,H,
   2 Hc_gmet,Hc_wmet,Hc_gurb,Hc_wurb,Rtop_met,Rtop_urb,
   3 KARMAN,Zmix,ORIrad,ORIdeg,WINDrad,k,beta,gamma,x,y,z

    INTEGER  WINDdeg

c  WIND10 is measured wind speed at ten-metre height (m s-1)
c  Uroof is wind speed at roof height (m s-1)
c  WINDdeg is wind direction (degrees)
c  WINDrad is wind direction (radians)
c  Uh is cross-canyon wind speed at roof level (m s-1)
c  Vh is down-canyon wind speed at roof level (m s-1)
c  H is average height of canyon walls (m)
c  W is canyon width (m)
c  Uxz is the horizontal component of the in-canyon vortex (m s-1)
c  Vz is the along canyon flow at height z in the canyon (m s-1)
c  Wxz is the vertical component of the in-canyon vortex (m s-1)
c  Ures_d is the resultant wind speed (m s-1)
c  Ustar_d is the friction velocity (m s-1)
c  KARMAN is the von Karman constant (0.4)
c  Zd is the zero-plane displacement (m)
c  Z0 is the roughness length (m)
c  Zmix is height of bottom of mixed layer (m)
```

```
c  ORIrad is direction of canyon axis (radians)
c  z is vertical coordinate of canyon cross-section (m)
c  x is horizontal coordinate of canyon cross-section (m)


c  Universal constants:
      KARMAN = 0.4
      PI = 3.1415926

c radiation heat transfer coefficient (for sol-air calculations only)

      Uroof =0.

      WINDrad = float(WINDdeg)*0.017453


c  calculate wind profile from standard wind data at met site:

      Zd_met = 0.001
      Z0_met = 0.01
      Zr_met = 10.0

      H=0.5*(H1_urb+H2_urb)
      Zmix = 2*H

      Ustar = WIND10*KARMAN/(log((Zr_met-Zd_met)/Z0_met))
      Umix  = Ustar/KARMAN*log((Zmix-Zd_met)/Z0_met)

c  Now calculate surface coefficient for met site:

        Usurf_met = Ustar/KARMAN*log((0.2-Zd_met)/Z0_met)

        Hc_gmet=6.42+3.96*Usurf_met
        Hc_wmet=Hc_gmet

c Calculate Hc for urban canyon:
c  When calculating Hc for the urban canyon surfaces, different
c  values used for ground and walls.


      ORIrad=ORIdeg*0.017453

c  First calculate wind at roof height from vertical profile.
c  Use approx. ratios for roughness length and displacement height:

      Zd_urb = 0.7*H
      Z0_urb = 0.1*H
      W = W_urb

      Uroof = Ustar/KARMAN*log((H-Zd_urb)/Z0_urb)
      Uh = Uroof*ABS(SIN(ORIrad-WINDrad))
      Vh = Uroof*ABS(COS(ORIrad-WINDrad))


c  The following calculations assume a rotor vortex occurs.
c  Wind speed is computed using Hotchkiss and Harlow model:

c  For wind at middle of canyon floor, set:

      z = 0.2
```

```
    x = W/2


    k = PI/W
    y = z-H
    beta = EXP(-2*k*H)
    gamma = EXP(k*y)

    Uxz = (Uh/(1-beta))*(gamma*(1+k*y)-beta*(1-k*y)/gamma)*SIN(k*x)
    Vz  = abs(Vh*(log(z)/log(H)))
    Wxz = (-Uh*k*y/(1-beta))*(gamma-(beta/gamma))*COS(k*x)
    Ures= sqrt(Uxz**2+Vz**2+Wxz**2)
    Usurf_urb=Ures

    Hc_gurb = 6.42+3.96*Ures

c For wind at middle of canyon walls, set:

    z = H/2
    x = 0.2


    k = PI/W
    y = z-H
    beta = EXP(-2*k*H)
    gamma = EXP(k*y)


    Uxz = (Uh/(1-beta))*(gamma*(1+k*y)-beta*(1-k*y)/gamma)*SIN(k*x)
    Vz  = abs(Vh*(log(z)/log(H)))
    Wxz = (-Uh*k*y/(1-beta))*(gamma-(beta/gamma))*COS(k*x)
    Ures= sqrt(Uxz**2+Vz**2+Wxz**2)

c  Hagishima and Tanimoto correlation for vertical walls:

    Hc_wurb = 4.47+10.2*ures


c  ----------------------------------------------------------------

c  The following sections attempts to model turbulent transfer or
c  resistance at canyon top. NOT CAT used in model yet.

c  Neutral drag coefficient (Garratt p.54, eq. 3.43):

c     Cdn=KARMAN**2/(log(Zmix/Z0_met))**2


c  From Garratt, p. 57:

c     Zt_met=z0_met/7.4
c     Zq_met=Zt_met

c     Chn=KARMAN**2/(log(Zmix/Zt_met))**2
c     Cqn=KARMAN**2/(log(Zmix/Zq_met))**2


c  calculate transfer resistance to mixed layer above canyon and
c reference site (see Barlow et al,2004 and Garratt, 1992)
```

```
c for reference site:

      IF(Ustar.GT.0.0) THEN
       Rtop_met=(Umix-Usurf_met)/Ustar**2.0
       Rtop_urb=(Umix-Usurf_urb)/Ustar**2.0
      ELSE
       Rtop_met=0.002
       Rtop_urb=0.002
      END IF

      RETURN
      END


c *******************************************************************

      SUBROUTINE PSA (H1,H2,W,H1off,H2off,W1off,W2off,
     I ALT,AZI,PI,Iglob,NORw1,PSAg,PSAw1,PSAw2)

c This subroutine calculates the partial shaded area of
c canyon surfaces. Canyon geometry allows input of different
c heights for the two canyon walls and for two adjacent buildings,
c each at a different (horizontal) offset from the canyon.


      INTEGER Iglob
      REAL ALT,AZI,PI,
     1 PSAg,PSAw1,PSAw2,H1,H2,W,NORw1,THETA

      LOGICAL SHADEw1,SHADEw2,SHADEadj1,SHADEadj2

c ALT is solar altitude (radians)
c AZI is solar azimuth (radians)
c Iglob is global solar radiation on horizontal surface (W m-2)
c PSAg is partial shaded area of the ground
c PSAwI is partial shaded area of wall 1
c PSAw2 is partial shaded area of wall 2
c W is canyon width (m)
c H1 is height of wall 1 (m)
c H2 is height of wall 2 (m)
c H1off is height of building adjacent to wall 1 (m)
c H2off is height of building adjacent to wall 2 (m)
c W1off is offset of building adjacent to wall 1 (m)
c W2off is offset of building adjacent to wall 2 (m)
c NORw1 is normal to wall I, the north-facing wall (radians)
c NORw2 is normal to wall 2, the south-facing wall (radians)
c ROOF1_shade is part of roof I shaded by adjacent building
c ROOF2_shade is part of roof 2 shaded by adjacent building


       THETA=ABS(COS(NORw1-AZI))

c THETA is the absolute value of the cosine of the angle
c between solar azimuth and the normal to the canyon wall.
c If AZI is represented as radians east or west of north,
c then ABS(COS(NORw1-AZI))=ABS(COS(NORw2-AZI)=THETA
c for all orientations.
```

```
c  First test to see if sun is above the horizon,
c  or if there is global radiation.

c  Then test to see which of the canyon walls
c  is shaded from direct solar radiation because of azimuth.
c
c   intialise variables
      SHADEadj1 = .FALSE.
      SHADEadj2 = .FALSE.
c
      IF((ALT.LE.0).or.(Iglob.EQ.0)) THEN
        PSAg=1
        PSAw1=1
        PSAw2=1
        GOTO 98
      ENDIF

      IF (ABS(NORw1-AZI).LT.PI/2) THEN
        SHADEw1=.FALSE.
        SHADEw2=.TRUE.
      ELSE
        SHADEw1=.TRUE.
        SHADEw2=.FALSE.
      ENDIF

c  Now test for shade by adjacent buildings

      ROOF1_shade=(H1off-H1)*THETA/TAN(ALT)
      IF (ROOF1_shade.GT.W1off) SHADEadj1=.TRUE.

      ROOF2_shade=(H2off-H2)*THETA/TAN(ALT)
      IF (ROOF2_shade.GT.W2off) SHADEadj2=.TRUE.

c  PSA of canyon floor:

      IF (SHADEw1) THEN
        IF (SHADEadj1) THEN
          PSAg=((H1off*THETA/TAN(ALT))-W1off)/W
        ELSE
          PSAg=(H1/W)*(THETA/TAN(ALT))
        ENDIF
      ELSE
        IF (SHADEadj2) THEN
          PSAg=((H2off*THETA/TAN(ALT))-W2off)/W
        ELSE
          PSAg=(H2/W)*(THETA/TAN(ALT))
        ENDIF
      ENDIF
      PSAg=ABS(PSAg)

c  PSA of wall 1:
c  (First check if it is shaded by azimuth)

      IF (SHADEw1) THEN
        PSAw1=1
      ELSE
        IF (.NOT.(SHADEw1).AND.(PSAg.GT.1)) THEN
          IF (SHADEadj1) THEN
            PSAw1=(H2off-(TAN(ALT)/THETA)*(W+W2off))/H1
          ELSE
```

```
        PSAw1=(H2-W*(TAN(ALT)/THETA))/H1
       ENDIF
      ELSE
       PSAw1=0
      ENDIF
     ENDIF
     PSAw1=ABS(PSAw1)


c PSA of wall 2:
c (First check if it is shaded by azimuth)

     IF (SHADEw2) THEN
      PSAw2=1
     ELSE
      IF (.NOT.(SHADEw2).AND.(PSAg.GT.1)) THEN
       IF (SHADEadj2) THEN
        PSAw2=(H1off-(TAN(ALT)/THETA)*(W+W1off))/H2
       ELSE
        PSAw2=(H1-W*(TAN(ALT)/THETA))/H2
       ENDIF
      ELSE
       PSAw2=0
      ENDIF
     ENDIF
     PSAw2=ABS(PSAw2)


     IF (PSAg.GT.1) PSAg=1
     IF (PSAw1.GT.1) PSAw1=1
     IF (PSAw2.GT.1) PSAw2=1

   98 RETURN
     END


c ------------------------------------------------------------------

     SUBROUTINE SENSIBLE (SITE,Ta,Qstar,Qstar_prev,dQstar,dQs,Qh,Qe,
    I alpha,A1,A2,A3,RAIN,RAIN_prev)

     REAL Qstar,dQstar,Qstar_prev,dQs,Qh,Qe,
    1 alpha,beta,gamma,s,es,A1,A2,A3,alphad,RAIN

     INTEGER SITE

     LOGICAL RAIN_prev

c This subroutine computes the storage flux (dQs) at a surface from
c the net radiant flux Qstar using the OHM model by Grimmond and Oke.
c The storage flux is then used to calculate the sensible heat flux
c from the surface, using the expression from Grimmond and Oke (2002).


c dQs is storage flux (W m-2)
c es is saturation vapour pressure (kPa)
c Qh is sensible heat flux (W m-2)
c Qstar is net radiant flux (W m-2)
c dQstar is the change in net radiant flux over the previous hour
c Qstar_prev is net radiant flux in the previous time step (W m-2)
```

```
c  s is the slope of the saturation vapour pressure vs. temperature
c  Ta is air dry bulb temperature (deg C)
c  alpha, beta and gamma are coefficients in the expression for Qh
c  A1,A2 and A3 are coefficients in the OHM model for storage flux
c  SITE is flag for reference or urban
c  RAIN is rainfall at weather station in current hour
c  RAIN_prev is TRUE if it rained (>0.5mm) in the previous hour
c  alphad is the dynamic rain-corrected value of alpha



      IF(SITE.EQ.4) THEN
       beta=10.0
      ELSE IF (SITE.EQ.7) THEN
       beta=3
      END IF

c      beta=20

c  gamma the psychometric constant (kPa K-1)

      gamma=0.065

c  Fist calculate storage flux (dQs):

      dQstar = Qstar-Qstar_prev

      dQs=a1*Qstar+a2*dQstar+a3


c  Now calculate atmospheric properties required to parcel
c  turbulent heat flux into sensible and latent components.

c  Saturation vapour pressure (kPa) is calculated using the Tetens formula:
c   (Tetens, 1930; as expressed by Murray, 1967)


      es=0.61078*EXP((17.269*Ta)/(Ta+237.29))

c 's', the slope of the saturation vapour pressure versus temperature
c curve:

      s = es*4099./(Ta+237.3)**2.

c Alternative formulation using Monteith equation 2.24.
c Units of gamma and s are in Pa K-1, not KPa K-1.

c      gamma=65
c      TaK=Ta+273.15
c      s=es*2439500*18.015/(8.31*TaK**2)

c Sensible heat flux is parameterised from net radiation and storage
c using revised format by Grimmond and Oke (2002).
c The value of alpha, the coefficient of moisture availability,
c is increased after rain, to model increased latent heat flux.

      IF(RAIN.GT.0.5.OR.RAIN_prev) THEN
       IF(SITE.EQ.4) THEN
        alphad=alpha+0.01
       ELSE IF (SITE.EQ.7) THEN
```

```
      alphad=alpha+0.10
     END IF
    ELSE
     alphad=alpha
    END IF

    Qh=(((1-alphad)+(gamma/s))/(1+(gamma/s)))*(Qstar-dQs)-beta

    Qe=(alphad/(1+(gamma/s)))*(Qstar-dQs)+beta

c  Reset value of alphad before next hourly period:

    alphad=alpha

    RETURN
    END

c -----------------------------------------------------------------------

    SUBROUTINE deltaT
   1 (Hsurf,Qh,dTa,MIX,SURFonW,Vol,MIXING,dTsol,Qe,dMC)

    REAL Hsurf,Qh,MIX(4),dTa,MIXING,SURFonW,dTsol

c  Following variables required for moisture calculations.
c  These are currently disabled.

c     REAL dTvap


c  This subroutine calculates the change in localized air temperature
c  resulting from the sensible heat flux at the surface, taking into
c  account empirical mixing factors to describe the effects of
c  buoyancy and mechanical turbulence near the surface.

    mix(1)=0.61
    mix(2)=0.76
    mix(3)=0.81
    mix(4)=0.92


     IF(dTsol.lt.-2.0) THEN
         mixing=mix(1)
       else if ((dTsol.lt.0.0).AND.(dTsol.ge.-2.0)) then
         mixing=mix(2)
       else if ((dTsol.lt.5.0).AND.(dTsol.ge.0.0)) then
         mixing=mix(3)
       else if (dTsol.gt.5) then
         mixing=mix(4)
     END IF


c     dTa=Qh*((1-mixing)/((SURFonW*Hsurf)+Htop))

    Rsurf=1/(SurfonW*Hsurf)
    Rtop=0.000
    Rtot=Rsurf+Rtop


    dTa=Qh*(1-mixing)*Rtot
```

```
c .........................................................
c The following section is NOT finished: speculates on methods
c to model changes in moisture content.
c .........................................................

c heat of vapourisation of water (KJ/kg)
   dTvap=2450
c density of air kg/m3
   dens = 1.2


c Qsurf is the transfer coefficient for moisture,
c analogous to Hsurf. Ratio depends on z0. Value of 1.3 is
c for z0=0.01m.

   Qsurf=Hsurf*1.3

   dMc=(3600*(1-mixing)*Qe/dTvap)/(Qsurf*Vol*dens*1000.)
c   dMc=((1-mixing)*Qe)/(Qsurf*VOL)

c End of speculative section.



   RETURN
   END

c ------------------------------------------------------------------

   SUBROUTINE SITEDATA
  1 (SITE,albed,W,H1,H2,W1off,H1off,W2off,H2off,
  2 ORIdeg,A1,A2,A3)

   REAL albed(2),H1,H2,W,H1off,H2off,W1off,W2off
   REAL A1(3),A2(3),A3(3)

   INTEGER SITE

c This subroutine reads data required for urban heat island effect
c calculations from input files met_site.txt or urb_site.txt

 60 FORMAT(60x,f7.2)

c   read past header and star line
   READ(SITE,*)
   READ(SITE,*)
   READ(SITE,*)

c Read urban cluster properties:


   READ(SITE,60) albed(1)
   READ(SITE,60) albed(2)
   READ(SITE,60) W
   READ(SITE,60) H1
   READ(SITE,60) H2
   READ(SITE,60) W1off
   READ(SITE,60) H1off
   READ(SITE,60) W2off
```

```
      READ(SITE,60) H2off
      READ(SITE,60) ORIdeg
c      READ(SITE,60) alpha
      READ(SITE,60) A1(1)
      READ(SITE,60) A2(1)
      READ(SITE,60) A3(1)
      READ(SITE,60) A1(2)
      READ(SITE,60) A2(2)
      READ(SITE,60) A3(2)
      READ(SITE,60) A1(3)
      READ(SITE,60) A2(3)
      READ(SITE,60) A3(3)


c     display data for checking

      write(6,61)albed(1)
61    format(' Mean surface albedo ground                ',f5.2)

      write(6,62)albed(2)
62    format(' Mean surface albedo walls                 ',f5.2)

      write(6,63)W
63    format(' Canyon width, in metres (W)               ',f5.2)

      write(6,64)H1
64    format(' Height of wall 1, in metres (H1)          ',f5.2)

      write(6,65)H2
65    format(' Height of wall 2, in metres (H2)          ',f5.2)

      write(6,66)W1off
66    format(' Offset of building adjacent to wall 1     ',f5.2)

      write(6,67)H1off
67    format(' Height of building adjacent to wall 1     ',f5.2)

      write(6,68)W2off
68    format(' Offset of building adjacent to wall 2     ',f5.2)

      write(6,69)H2off
69    format(' Height of building adjacent to wall 2     ',f5.2)

c      write(6,70)alpha
c 70   format(' Empirical coefficient for soil moisture   ',f5.2)

      write(6,71)A1(1)
71    format(' Heat storage coefficient A1 ground        ',f5.2)

      write(6,72)A2(1)
72    format(' Heat storage coefficient A2 ground        ',f5.2)

      write(6,73)A3(1)
73    format(' Heat storage coefficient A3 ground        ',f5.1)

      write(6,74)A1(2)
74    format(' Heat storage coefficient A1 wall 1        ',f5.2)

      write(6,75)A2(2)
75    format(' Heat storage coefficient A2 wall 1        ',f5.2)
```

```
      write(6,76)A3(2)
76    format(' Heat storage coefficient A3 wall 1          ',f5.1)

      write(6,77)A1(3)
77    format(' Heat storage coefficient A1 wall 2          ',f5.2)

      write(6,78)A2(3)
78    format(' Heat storage coefficient A2 wall 2          ',f5.2)

      write(6,79)A3(3)
79    format(' Heat storage coefficient A3 wall 2          ',f5.1)


      pause '  Check data, then press ENTER to continue.'

      RETURN
      END


c ------------------------------------------------------------------

       subroutine moisture (pt,t,rh,w)

       REAL pt,t,w,ps,p,psw
       INTEGER rh

c  This subroutine calculates the moisture content of air from
c  dry bulb temperature and relative humidity.

c  (Total atmospheric pressure also required - if not available,
c  use Pt=1013.25 mb.)

c  p is actual pressure of water vapour (kPa)
c  ps is saturated pressure of water vapour (kPa)
c  psw is saturated pressure of water vapour at the TWB (kPa)
c  pt is total atmospheric pressure (kPa)
c  rh is relative humidity (%)
c  t is dry bulb temperature (deg C)
c  w is moisture content of air (g of water per kg of dry air)

c  TWB - thermodynamic wet bulb temperature


        ps = psw(t)
c         PRINT*,' ps (mhar) = ',ps
        P = 0.01*rh*ps

        FS=-7.3E-06*t+1.00444
        IF(t.GE.11.0.AND.t.LT.26.0) FS=1.32E-05*t+1.004205
        IF(t.GE.26.0) FS=4.05E-05*t+1.003497

        w = 1000*0.622*FS*p/(pt-FS*p)

        return
        end
c ------------------------------------------------------------------


      SUBROUTINE MOMENT(DATA,N,AVE,ADEV,SDEV,VAR,SKEW,CURT)
```

```
      DIMENSION DATA(N)
      IF(N.LE.1)PAUSE 'N must be at least 2'
      S=0.
      DO 11 J=1,N
        S=S+DATA(J)
11    CONTINUE
      AVE=S/N
      ADEV=0.
      VAR=0.
      SKEW=0.
      CURT=0.
      DO 12 J=1,N
        S=DATA(J)-AVE
        ADEV=ADEV+ABS(S)
        P=S*S
        VAR=VAR+P
        P=P*S
        SKEW=SKEW+P
        P=P*S
        CURT=CURT+P
12    CONTINUE
      ADEV=ADEV/N
      VAR=VAR/(N-1)
      SDEV=SQRT(VAR)
      IF(VAR.GT.1E-08)THEN
        SKEW=SKEW/(N*SDEV**3)
        CURT=CURT/(N*VAR**2)-3.
      ELSE
        PAUSE 'no skew or kurtosis when zero variance'
      ENDIF
      RETURN
      END
c
c *********************************
c
      SUBROUTINE FIT(X,Y,NDATA,SIG,MWT,A,B,SIGA,SIGB,CHI2,Q)
      DIMENSION X(NDATA),Y(NDATA),SIG(NDATA)
c
c      PRINT*, ndata
c
      SX=0.
      SY=0.
      ST2=0.
      B=0.
      IF(MWT.NE.0) THEN
        SS=0.
        DO 11 I=1,NDATA
          WT=1./(SIG(I)**2)
          SS=SS+WT
          SX=SX+X(I)*WT
          SY=SY+Y(I)*WT
11      CONTINUE
      ELSE
        DO 12 I=1,NDATA
          SX=SX+X(I)
          SY=SY+Y(I)
12      CONTINUE
        SS=FLOAT(NDATA)
      ENDIF
      SXOSS=SX/SS
```

```
      IF(MWT.NE.0) THEN
        DO 13 I=1,NDATA
          T=(X(I)-SXOSS)/SIG(I)
          ST2=ST2+T*T
          B=B+T*Y(I)/SIG(I)
13      CONTINUE
      ELSE
        DO 14 I=1,NDATA
          T=X(I)-SXOSS
          ST2=ST2+T*T
          B=B+T*Y(I)
14      CONTINUE
      ENDIF
      B=B/ST2
      A=(SY-SX*B)/SS
      SIGA=SQRT((1.+SX*SX/(SS*ST2))/SS)
      SIGB=SQRT(1./ST2)
      CHI2=0.
      IF(MWT.EQ.0) THEN
        DO 15 I=1,NDATA
          CHI2=CHI2+(Y(I)-A-B*X(I))**2
15      CONTINUE
        Q=1.
        SIGDAT=SQRT(CHI2/(NDATA-2))
        SIGA=SIGA*SIGDAT
        SIGB=SIGB*SIGDAT
      ELSE
        DO 16 I=1,NDATA
          CHI2=CHI2+((Y(I)-A-B*X(I))/SIG(I))**2
16      CONTINUE
        Q=GAMMQ(0.5*(NDATA-2),0.5*CHI2)
      ENDIF
      RETURN
      END
c
      FUNCTION GAMMQ(A,X)
      IF(X.LT.0..OR.A.LE.0.)PAUSE
      IF(X.LT.A+1.)THEN
        CALL GSER(GAMSER,A,X,GLN)
        GAMMQ=1.-GAMSER
      ELSE
        CALL GCF(GAMMCF,A,X,GLN)
        GAMMQ=GAMMCF
      ENDIF
      RETURN
      END
c
      SUBROUTINE GSER(GAMSER,A,X,GLN)
      PARAMETER (ITMAX=100,EPS=3.E-7)
      GLN=GAMMLN(A)
      IF(X.LE.0.)THEN
        IF(X.LT.0.)PAUSE
        GAMSER=0.
        RETURN
      ENDIF
      AP=A
      SUM=1./A
      DEL=SUM
      DO 11 N=1,ITMAX
        AP=AP+1.
```

```
      DEL=DEL*X/AP
      SUM=SUM+DEL
      IF(ABS(DEL).LT.ABS(SUM)*EPS)GO TO 1
11    CONTINUE
     PAUSE 'A too large, ITMAX too small'
1    GAMSER=SUM*EXP(-X+A*LOG(X)-GLN)
     RETURN
     END
c
     SUBROUTINE GCF(GAMMCF,A,X,GLN)
     PARAMETER (ITMAX=100,EPS=3.E-7)
     GLN=GAMMLN(A)
     GOLD=0.
     A0=1.
     A1=X
     B0=0.
     B1=1.
     FAC=1.
     DO 11 N=1,ITMAX
      AN=FLOAT(N)
      ANA=AN-A
      A0=(A1+A0*ANA)*FAC
      B0=(B1+B0*ANA)*FAC
      ANF=AN*FAC
      A1=X*A0+ANF*A1
      B1=X*B0+ANF*B1
      IF(INT(A1).NE.0)THEN
       FAC=1./A1
       G=B1*FAC
       IF(ABS((G-GOLD)/G).LT.EPS)GO TO 1
       GOLD=G
      ENDIF
11    CONTINUE
     PAUSE 'A too large, ITMAX too small'
1    GAMMCF=EXP(-X+A*ALOG(X)-GLN)*G
     RETURN
     END
c
     FUNCTION GAMMLN(XX)
     REAL*8 COF(6),STP,HALF,ONE,FPF,X,TMP,SER
     DATA COF,STP/76.18009173D0,-86.50532033D0,24.01409822D0,
    1    -1.231739516D0,.120858003D-2,-.536382D-5,2.50662827465D0/
     DATA HALF,ONE,FPF/0.5D0,1.0D0,5.5D0/
     X=XX-ONE
     TMP=X+FPF
     TMP=(X+HALF)*LOG(TMP)-TMP
     SER=ONE
     DO 11 J=1,6
      X=X+ONE
      SER=SER+COF(J)/X
11    CONTINUE
     GAMMLN=TMP+LOG(STP*SER)
     RETURN
     END
c
c *********************************************
     subroutine WILLMOTT(P,O,Ndata,A,B,RMSE,RMSEs,RMSEu,d,
    1 MSE,MSEs,MSEu)
c
     REAL P(9000),O(9000),PHat(9000),MSE,Pd(9000),Od(9000),MSEs,MSEu
```

```
c
c  initialize SUMs
      SUM1=0.
      SUM2=0.
      SUM3=0.
      SUMm=0.
      PE=0.
c  calcuate Mean of O
      do 5 i=1,Ndata
      SUMm = SUMm + O(i)
5     continue
      Omean = SUMm/Ndata
c     PRINT*, Omean
c
c  calculate PHat
c
      do 10 i=1,Ndata
      PHat(i) = A + B*O(i)
10    continue
c

      do 20 i=1,Ndata
      SUM1 = SUM1+ (PHat(i) - O(i))**2
20    continue
      MSEs  = SUM1/Ndata
      RMSEs = SQRT(SUM1/Ndata)
c
      do 30 i=1,Ndata
      SUM2 = SUM2+ (P(i) - PHat(i))**2
30    continue
      MSEu = SUM2/Ndata
      RMSEu = SQRT(SUM2/Ndata)
c
      do 40 i=1,Ndata
      SUM3 = SUM3+ (P(i) - O(i))**2
40    continue
c     PRINT*,sum3
      MSE = SUM3/Ndata
      RMSE = SQRT(MSE)
c
      do 50 i=1,Ndata
      Pd(i) = P(i) - Omean
50    continue
c
      do 60 i=1,Ndata
      Od(i) = O(i) -Omean
60    continue
c
      do 70 i=1,Ndata
      PE = PE + (ABS(Pd(i)) +ABS(Od(i)))**2
70    continue
c
      d = 1. - Ndata*MSE/PE
c
      RETURN
      END


c ******************************************
```

```
      subroutine CONFIRM(P,O,V,Ndata,U,D)

      REAL P(9000),O(9000),V(9000)


c  initialise sum variables
      SUMp =0.
      SUMo =0.
      SUMv =0.
      SUM1 =0.
      SUM2 =0.

      do 10 i=1,Ndata
      SUMp = SUMp + P(i)**2.
10    continue
      SUMp =SUMp/Ndata

      do 20 i=1,Ndata
      SUMo = SUMo + O(i)**2.
20    continue
      SUMo = SUMo/Ndata

      do 30 i=1,Ndata
      SUMv = SUMv + V(i)**2.
30    continue
      SUMv = SUMv/Ndata

      do 40 i=1,Ndata
      SUM1 = SUM1+ (P(i) - O(i))**2.
40    continue

c  inequality coefficieny (U(p,o))
      aNUM1 = SQRT(SUM1/Ndata)
      U = aNUM1/(SQRT(SUMp)+SQRT(SUMo))

      do 50 i=1,Ndata
      SUM2 = SUM2 + (O(i) - V(i))**2.
50    continue

c inequality coefficient (U(o,v))
      aNUM2 =SQRT(SUM2/Ndata)
      Uov = aNUM2/(SQRT(SUMo)+SQRT(SUMv))

c  Confirmation co-efficient (U(o,v)-U(p,o))/U(o,v)
      D = (Uov-U)/Uov

      RETURN
      END


c  -----------------------------------------------------------

      subroutine TandW(t,w,rh)

      real pt,t,w,ps,p
      integer rh


c  Calculate RH (and other parameters) from dry bulb temperature t (C)
c  and moisture content of air w (g of water/kg of dry air)
```

```
c ***********************
c  Note: w given in grams!
c ***********************

c  Total atmospheric pressure:
      pt=1013.25


c     h = 1.0048*t+(1.8003*t+2502.68)*w;
c     v = 6.89476*0.062428*(491.7+1.8*t)*(0.622+w)/(1.679*pt);
c     tw = WB(pt,t,w);
c     psw = p1(tw);
c     ws = 0.622*psw/(pt-psw);
c     td = DP(p);


      p =  w*1E-03*pt/(0.622+w*1E-03)
      ps = psw(t)
      rh = INT(100.0*p/ps)

      end
c
      FUNCTION psw(t)

c calculate saturated pressure of water vapour (psw) at temperature t
c psw mbar (hPa)
c
       x = 673.4 - 1.8*DBLE(t)
       psw = (6.89476*3206.18/exp(2.302585*x*(3.2437814+3.26014E-3*x
+2.00658E-9*x**3.)/((1165.09-x)*(1.0+1.21547e-3*x))))*10.

       return
       end
```

NOTE:
Canyon Air Temperature v.1.0
CD-ROM is included with the print
copy of the thesis held in the
University of Adelaide Library