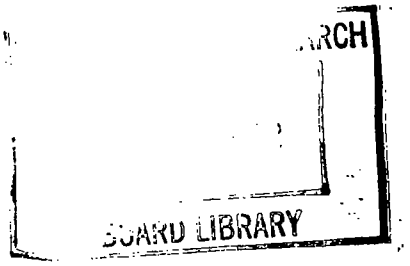

SHRP-P-635

Analysis of Section Homogeneity, Non-Representative Test Pit and Section Data, and Structural Capacity

FWDCHECK Version 2.00

Volume III—Program Listing

Strategic Highway Research Program
National Research Council

Strategic Highway Research Program Executive Committee

John R. Tabb, Chairman
Mississippi Highway Department

William G. Agnew
General Motors Research (retired)

E. Dean Carlson, *ex officio*
Federal Highway Administration

A. Ray Chamberlain
Colorado Department of Highways

Michael J. Cuddy
New York Department of Transportation

Raymond F. Decker
University Science Partners Inc.

Thomas B. Deen, *ex officio*
Transportation Research Board

Thomas M. Downs
New Jersey Department of Transportation

Francis B. Francois, *ex officio*
American Association of State Highway and Transportation Officials

William L. Giles
Ruan Transportation Management Systems

Jack S. Hodge
Virginia Department of Transportation

Donald W. Lucas
Indiana Department of Transportation

Harold L. Michael
Purdue University

Wayne Muri
Missouri Highway and Transportation Department

M. Lee Powell, III
Ballenger Paving Company, Inc.

Henry A. Thomason, Jr.
Texas Department of Transportation

Stanley I. Warshaw
National Institute of Standards and Technology

Roger L. Yarbrough
Apcon Corporation

Key SHRP Staff

Damian J. Kulash
Executive Director

Edward T. Harrigan
Asphalt Program Manager

Kathryn Harrington-Hughes
Communications Director

Don M. Harriott
Concrete & Structures/Highway Operations Program Manager

Harry Jones
Finance & Administration Director

Guy W. Hager
Implementation Manager

SHRP-P-635

Analysis of Section Homogeneity, Non-Representative Test Pit and Section Data, and Structural Capacity

FWDCHECK Version 2.00

Volume III—Program Listing

P-001B Technical Advisory Staff
PCS/Law Engineering

Strategic Highway Research Program
National Research Council
Washington, DC 1993

SHRP–P–635
Contract P-001B

Program Manager: *Neil Hawks*
Project Manager: *Cheryl Richter*
Production Editor: *Marsha Barrett*
Program Area Secretary: *Carina Hreib*

June 1993

key words:
deflection testing
falling weight deflectometer
quality assurance
structural evaluation

Strategic Highway Research Program
National Academy of Sciences
2101 Constitution Avenue N.W.
Washington, DC 20418

(202) 334-3774

350/NAP/693

Acknowledgments

The research described herein was supported by the Strategic Highway Research Program (SHRP). SHRP is a unit of the National Research Council that was authorized by section 128 of the Surface Transportation and Uniform Relocation Assistance Act of 1987.

ABSTRACT

Nondestructive deflection testing using falling weight deflectometers is one element of the monitoring effort currently underway by the Strategic Highway Research Program (SHRP) for the Long Term Pavement Performance (LTPP) study. Because accurate data is key to the success of the LTPP study, SHRP has implemented a number of measures to ensure the quality of deflection data. They include equipment comparison and calibration, standardized field testing procedures and field data checks, and quality assurance software.

Equipment calibration and field data checks built into the FWD data acquisition software are the first line of defense against invalid deflection data. The second line of defense is a computer program, called FWDSCAN, which verifies the integrity, completeness, and compliance with the established test pattern of the field data after it is delivered to the SHRP regional office. For the final stage in the quality assurance process, a computer program called FWDCHECK has been developed to analyze deflection data for test section homogeneity, the degree to which test pit data is representative of the section, the presence of data outliers within the section, and overall reasonableness from a structural capacity viewpoint.

This report focuses on the FWDCHECK program. The report is provided in three separate volumes: Technical Documentation, User's Guide, and Program Listing. The technical documentation gives a detailed description of the program including the analyses and algorithms used. A detailed description of the program usage is provided in the User's Guide. Finally, a complete printout of the computer source code is included in the third volume, Program Listing.

```
DECLARE SUB PlotSetup (XMin!, XMax!, YMin!, YMax!, PlotXMin!, PlotXMax!, PlotYMin!, PlotYMax!, LX!, XScale!)
DECLARE SUB RedimArrays ()
DECLARE SUB ResetVariables ()
DECLARE SUB PlotDeviationCurve (PlotLocation%, PlotHeight%, PlotDeflector%, PlotYMin!, PlotYMax!)
DECLARE SUB PlotNormDeflCurves (PlotLocation%, PlotHeight%, PlotYMax!)
DECLARE SUB CalcStationStats (StatsFailed%)
DECLARE SUB CalcOverallStats ()
DECLARE SUB CalcTPOverallStats ()
DECLARE SUB PlotCorrDeflCurves (PlotLocation%, PlotHeight%, PlotYMax!)
DECLARE SUB SetMaterialInfo ()
DECLARE SUB OutliersMenu (ExitCode%)
DECLARE SUB SubsectionAnalysisMenu (ExitCode%)
DECLARE SUB CheckHeader2 (ExitCode%)
DECLARE SUB ReadFWDDataFile ()
DECLARE SUB ReadNextLine (DataType%)
DECLARE SUB GetFileName (ExitCode%)
DECLARE SUB StatisticsMenu (ExitCode%)
DECLARE SUB StructuralMenu (ExitCode%)
DECLARE SUB ReadPeaks2 ()
DECLARE SUB DisplayCopyright ()
DECLARE FUNCTION DeflectionRatio! (E1s!, E1f!)
DECLARE FUNCTION StandardDeviation! (SSum!, SSum2!, ItemCount%) 'do not erase
DECLARE FUNCTION Beta1! (a!, B!, X!)                            'do not erase
DECLARE FUNCTION BetaCF! (a!, B!, X!)                           'do not erase
DECLARE FUNCTION GammaLN! (XX!)                                 'do not erase
DECLARE FUNCTION Log10 (Value!)                                 'do not erase
'$INCLUDE: 'declare.inc'
'$INCLUDE: 'fdeclar1.inc'
'$INCLUDE: 'fdeclar2.inc'
'$INCLUDE: 'cmnblank.inc'
'$INCLUDE: 'fwdcheck.inc'

CONST True% = -1, False% = 0, MaxNumStations% = 50, MaxLocations% = 6
CONST MaxDeflectors% = 7, MaxNumPeaks% = 16, MaxHeights% = 4, MaxLayers% = 10
CONST MinPointsInSubSection% = 4, ThresholdAlpha = 95, MaxTimes% = 10

GP.Monitor% = Monitor%
CALL DisplayCopyright
CALL SetMaterialInfo
FPath$ = ""
Page% = 1: MaxPage% = 5
DO
  SELECT CASE Page%
    CASE 1
      CALL RedimArrays
      CALL ResetVariables
      SCREEN 0: WIDTH 80, 25: CLS
      CALL GetFileName(ExitCode%)
      Source$ = FPath$ + File$ + Ext$
      OPEN Source$ FOR INPUT AS #1
      Output$ = FPath$ + File$ + ".RES"
      DBRoot$ = LEFT$(File$, 3)         'kludge to show section ID on screen
      WDBExt$ = MID$(File$, 4, 3)       'kludge to show section ID on screen
      CLS
      LineCounter& = 0
      LS% = LEN(Source$)
      Msg$ = "Reading FWD data from file: "
      IF LS% > 51 THEN
        c1% = 41 - LS% \ 2
        LOCATE 9, 27: PRINT Msg$
        LOCATE 10, c1%: PRINT Source$
      ELSE
        c1% = 41 - (LS% + 28) \ 2
        LOCATE 10, c1%: PRINT Msg$; Source$
      END IF
      CALL ReadFWDDataFile
      CLOSE #1
      IF (LocationList%(4) AND NOT LocationList%(3)) OR (LocationList%(2) AND NOT LocationList%(1)) THEN
        Found% = False%
        FOR ss% = 0 TO 1
          File$ = LEFT$(File$, 7) + QPTrim$(STR$(ss%))
          Source$ = FPath$ + File$ + Ext$
          IF Exist%(Source$) THEN
```

```
          OPEN Source$ FOR INPUT AS #1
          CLS
          LineCounter& = 0
          LS% = LEN(Source$)
          Msg$ = "Reading FWD test pit data from file: "
          IF LS% > 41 THEN
            c1% = 41 - LS% \ 2
            LOCATE 9, 23: PRINT Msg$
            LOCATE 10, c1%: PRINT Source$
          ELSE
            c1% = 41 - (LS% + 37) \ 2
            LOCATE 10, c1%: PRINT Msg$; Source$
          END IF
          DO
            IF LineCounter& < 37 THEN
              CALL CheckHeader2(ExitCode%)
              IF ExitCode% = 999 THEN EXIT DO
            ELSE
              CALL ReadNextLine(DataType%)
              SELECT CASE DataType%
                CASE 1                          'peak deflection data block
                  CALL ReadPeaks2
                CASE -1
                  EXIT DO
                CASE ELSE
                  'do nothing
              END SELECT
            END IF
          LOOP
          CLOSE #1
        END IF
        IF NumTPStations% > 0 THEN Found% = True%: EXIT FOR
      NEXT ss%
      IF NOT Found% THEN
        REDIM PUText$(1)
        PUText$(1) = "Test pit data NOT found where expected.  Proceeding without it..."
        CALL PopupWarning(0, 0, "")
      END IF
    END IF
    FOR I% = 1 TO NumTPStations%
      IF TPStation%(I%) > 0 AND TPStation%(I%) < 100 THEN TPStation%(I%) = -TPStation%(I%)
    NEXT I%
    CALL CalcStationStats(StatsFailed%)
    IF NOT StatsFailed% THEN
      CALL CalcOverallStats
      CALL CalcTPOverallStats
      IF English% THEN
        RadFactor = 1
        LoadFactor = 1                                  'pounds to pounds
        DeflFactor = .001                               'mils to inches
      ELSE
        RadFactor = 1 / 25.4
        LoadFactor = .145 * 3.141593 * (LPR / 25.4) ^ 2   'kPa to pounds
        DeflFactor = .001 / 25.4                        'microns to inches
      END IF
      TLPR = LPR * RadFactor
      OPEN Output$ FOR OUTPUT AS #2
    ELSE          'bad deflection data found!
      ExitCode% = -2        'force selection of new file
    END IF
  CASE 2
    CALL StatisticsMenu(ExitCode%)
  CASE 3
    CALL SubsectionAnalysisMenu(ExitCode%)
  CASE 4
    CALL OutliersMenu(ExitCode%)
  CASE 5
    CALL StructuralMenu(ExitCode%)
END SELECT
SELECT CASE ExitCode%
  CASE 1
    Page% = Page% + 1
  CASE -1
```

```
      Page% = Page% - 1
    CASE -2
      Page% = 1 'Select a new file
    CASE 0
      'nothing yet
  END SELECT
  IF Page% < 1 THEN Page% = 1
  IF Page% > MaxPage% THEN Page% = 1
LOOP
END

SUB CalcCorrStationStats STATIC
' Calc average normalized corrected deflection for each drop height at each station:
  REDIM MeanCorrDefl(NumStations%, MaxHeight%), TPMeanCorrDefl(2, MaxHeight%)
  FOR I% = 1 TO NumStations%
    Sum = 0
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF TempCorrDefl(I%, J%) > 0 THEN
          Sum = Sum + TempCorrDefl(I%, J%) / TestLoad(I%, J%)
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          MeanCorrDefl(I%, LastHeight%) = Sum / ACount%
          Sum = TempCorrDefl(I%, J%) / TestLoad(I%, J%)
        END IF
        LastHeight% = Heights%(J%)
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          MeanCorrDefl(I%, LastHeight%) = Sum / ACount%
        END IF
      END IF
    NEXT J%
  NEXT I%
' Calc average normalized corrected deflection for each drop height at each test pit:
  FOR I% = 1 TO NumTPStations%
    Sum = 0
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF TPTempCorrDefl(I%, J%) > 0 THEN
          Sum = Sum + TPTempCorrDefl(I%, J%) / TPTestLoad(I%, J%)
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          TPMeanCorrDefl(I%, LastHeight%) = Sum / ACount%
          Sum = TPTempCorrDefl(I%, J%) / TPTestLoad(I%, J%)
        END IF
        LastHeight% = Heights%(J%)
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          TPMeanCorrDefl(I%, LastHeight%) = Sum / ACount%
        END IF
      END IF
    NEXT J%
  NEXT I%
END SUB

SUB CalcMidDepthTemps STATIC
  CALL HiliteColor
  LOCATE 22, 11: PRINT "calculating asphalt mid depth temperatures.  please wait..."
  REDIM MidTemp(10, 2), Gradient(10, 2), AdjMeasMin%(10, 2)
  REDIM AdjFWDMin%(MaxNumStations%), TPAdjFWDMin%(2)
  IF CrossingMidnight% THEN
```

```
    IF FWDMinutes%(1) < 720 THEN
      AdjFWDMin%(1) = FWDMinutes%(1) + 1440
    ELSE
      AdjFWDMin%(1) = FWDMinutes%(1)
    END IF
  ELSE
    AdjFWDMin%(1) = FWDMinutes%(1)
  END IF
  FOR I% = 2 TO NumStations%            'makes FWD times ABSOLUTELY sequential
    IF CrossingMidnight% THEN
      IF AdjFWDMin%(I% - 1) > FWDMinutes%(I%) THEN  'after crossing midnight
        AdjFWDMin%(I%) = FWDMinutes%(I%) + 1440
      ELSE
        AdjFWDMin%(I%) = FWDMinutes%(I%)
      END IF
    ELSE
      AdjFWDMin%(I%) = FWDMinutes%(I%)
    END IF
  NEXT I%
  IF CrossingMidnight% THEN
    IF TPFWDMinutes%(1) < 720 THEN
      TPAdjFWDMin%(1) = TPFWDMinutes%(1) + 1440
    ELSE
      TPAdjFWDMin%(1) = TPFWDMinutes%(1)
    END IF
  ELSE
    TPAdjFWDMin%(1) = TPFWDMinutes%(1)
  END IF
  FOR I% = 2 TO TPNumStations%          'makes FWD times ABSOLUTELY sequential
    IF CrossingMidnight% THEN
      IF TPAdjFWDMin%(I% - 1) > TPFWDMinutes%(I%) THEN  'after crossing midnight
        TPAdjFWDMin%(I%) = TPFWDMinutes%(I%) + 1440
      ELSE
        TPAdjFWDMin%(I%) = TPFWDMinutes%(I%)
      END IF
    ELSE
      TPAdjFWDMin%(I%) = TPFWDMinutes%(I%)
    END IF
  NEXT I%
  FOR J% = 1 TO 2                                  'makes TEMPERATURE times
    AdjMeasMin%(1, J%) = MeasMinutes%(1, J%)       ' ABSOLUTELY sequential
    FOR I% = 2 TO NumTimes%(J%)
      IF CrossingMidnight% THEN
        IF AdjMeasMin%(I% - 1, J%) > MeasMinutes%(I%, J%) THEN 'after crossing midnight
          AdjMeasMin%(I%, J%) = MeasMinutes%(I%, J%) + 1440
        ELSE
          AdjMeasMin%(I%, J%) = MeasMinutes%(I%, J%)
        END IF
      ELSE
        AdjMeasMin%(I%, J%) = MeasMinutes%(I%, J%)
      END IF
    NEXT I%
  NEXT J%
  FOR J% = 1 TO 2
    FOR I% = 1 TO NumTimes%(J%)
      Sum = 0
      FOR K% = 1 TO NumDepths%
        Sum = Sum + TimeTemp(I%, J%, K%)
      NEXT K%
      MidTemp(I%, J%) = Sum / NumDepths%
    NEXT I%
    FOR Interval% = 1 TO NumTimes%(J%) - 1
      Gradient(Interval%, J%) = (MidTemp(Interval% + 1, J%) - MidTemp(Interval%, J%)) / (AdjMeasMin%(Interval% + 1, J%) -
AdjMeasMin%(Interval%, J%))
    NEXT Interval%
  NEXT J%
  FOR I% = 1 TO NumStations%
    IF Station%(I%) < 250 THEN J% = 1 ELSE J% = 2
    LoIndex% = 1      'Gradient = temp gradient with INCREASING time
    HiIndex% = 2
    TimeDiff% = AdjFWDMin%(I%) - AdjMeasMin%(LoIndex%, J%)
    IF TimeDiff% < 0 THEN     'Gradient w/+time, before lotime uses lotemp
      MidDepthTemp(I%) = MidTemp(LoIndex%, J%)
```

```
    ELSEIF AdjFWDMin%(I%) < AdjMeasMin%(HiIndex%, J%) THEN
      MidDepthTemp(I%) = Gradient(LoIndex%, J%) * TimeDiff% + MidTemp(LoIndex%, J%)
    ELSE
      DO
        IF HiIndex% < NumTimes%(J%) THEN
          LoIndex% = LoIndex% + 1
          HiIndex% = HiIndex% + 1
          TimeDiff% = AdjFWDMin%(I%) - AdjMeasMin%(LoIndex%, J%)
          IF TimeDiff% > 0 THEN     'Gradient w/+time, -time
            IF AdjFWDMin%(I%) < AdjMeasMin%(HiIndex%, J%) THEN
              MidDepthTemp(I%) = Gradient(LoIndex%, J%) * TimeDiff% + MidTemp(LoIndex%, J%)
              EXIT DO
            END IF
          END IF
        ELSE                    'after last temp measurement
          MidDepthTemp(I%) = MidTemp(HiIndex%, J%)
          EXIT DO
        END IF
      LOOP
    END IF
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    IF TPStation%(I%) < 250 THEN J% = 1 ELSE J% = 2
    LoIndex% = 1      'Gradient = temp gradient with INCREASING time
    HiIndex% = 2
    TimeDiff% = TPAdjFWDMin%(I%) - AdjMeasMin%(LoIndex%, J%)
    IF TimeDiff% < 0 THEN     'Gradient w/+time, before lotime uses lotemp
      TPMidDepthTemp(I%) = MidTemp(LoIndex%, J%)
    ELSEIF TPAdjFWDMin%(I%) < AdjMeasMin%(HiIndex%, J%) THEN
      TPMidDepthTemp(I%) = Gradient(LoIndex%, J%) * TimeDiff% + MidTemp(LoIndex%, J%)
    ELSE
      DO
        IF HiIndex% < NumTimes%(J%) THEN
          LoIndex% = LoIndex% + 1
          HiIndex% = HiIndex% + 1
          TimeDiff% = TPAdjFWDMin%(I%) - AdjMeasMin%(LoIndex%, J%)
          IF TimeDiff% > 0 THEN     'Gradient w/+time, -time
            IF TPAdjFWDMin%(I%) < AdjMeasMin%(HiIndex%, J%) THEN
              TPMidDepthTemp(I%) = Gradient(LoIndex%, J%) * TimeDiff% + MidTemp(LoIndex%, J%)
              EXIT DO
            END IF
          END IF
        ELSE                    'after hitime uses hitemp
          TPMidDepthTemp(I%) = MidTemp(HiIndex%, J%)
          EXIT DO
        END IF
      LOOP
    END IF
  NEXT I%
  ERASE MidTemp, Gradient, AdjMeasMin%, AdjFWDMin%, TPAdjFWDMin%
END SUB

SUB CalcOutliers STATIC
  REDIM Deviation(NumStations%, MaxHeight%, NumDeflectors%)
  REDIM TPDeviation(NumTPStations%, MaxHeight%, NumDeflectors%)
  LastStation% = 0
  FOR L% = 1 TO NumSubSect%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(L%) + FirstStation% - 1
    FOR I% = FirstStation% TO LastStation%  'calc deviation of norm deflection for each height and sensor
      PL% = PointLocation%(I%) + 1
      FOR J% = 1 TO MaxHeight%
        IF SubSectionMeanDefl(PL%, J%, 1, L%) <> 0 AND SubSectionStDevDefl(PL%, J%, 1, L%) <> 0 THEN
          FOR K% = 1 TO NumDeflectors%
            IF K% = 1 THEN
              Deviation(I%, J%, K%) = (MeanCorrDefl(I%, J%) - SubSectionMeanDefl(PL%, J%, K%, L%)) / SubSectionStDevDefl(PL%,
J%, K%, L%)
            ELSE
              Deviation(I%, J%, K%) = (MeanNormDefl(I%, J%, K%) - SubSectionMeanDefl(PL%, J%, K%, L%)) /
SubSectionStDevDefl(PL%, J%, K%, L%)
            END IF
          NEXT K%
        ELSE
```

```
        FOR K% = 1 TO NumDeflectors%
          Deviation(I%, J%, K%) = 0
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
NEXT L%
FOR I% = 1 TO NumTPStations%        'for all test pit locations (1 or 2)
  FOR J% = 1 TO MaxHeight%
    IF TPStation%(I%) <= 0 THEN L% = 1 ELSE L% = NumSubSect%
    Mean1 = SubSectionMeanDefl(PL%, J%, 1, L%)
    StDev1 = SubSectionStDevDefl(PL%, J%, 1, L%)
    IF Mean1 <> 0 AND StDev1 <> 0 THEN
      FOR K% = 1 TO NumDeflectors%
        IF K% = 1 THEN
          TPDeviation(I%, J%, K%) = (TPMeanCorrDefl(I%, J%) - SubSectionMeanDefl(PL%, J%, K%, L%)) /
SubSectionStDevDefl(PL%, J%, K%, L%)
        ELSE
          TPDeviation(I%, J%, K%) = (TPMeanNormDefl(I%, J%, K%) - SubSectionMeanDefl(PL%, J%, K%, L%)) /
SubSectionStDevDefl(PL%, J%, K%, L%)
        END IF
      NEXT K%
    ELSE
      FOR K% = 1 TO NumDeflectors%
        TPDeviation(I%, J%, K%) = 0
      NEXT K%
    END IF
  NEXT J%
NEXT I%
END SUB

SUB CalcOverallStats STATIC
' Calc pass avg & std dev of norm deflection for each height and sensor:
  REDIM PassSum(MaxLocations%, MaxHeight%, NumDeflectors%)
  REDIM PassSumSq(MaxLocations%, MaxHeight%, NumDeflectors%)
  REDIM OverallMeanNormDefl(MaxLocations%, MaxHeight%, NumDeflectors%)
  REDIM OverallStDevNormDefl(MaxLocations%, MaxHeight%, NumDeflectors%)
  REDIM OverallCVNormDefl(MaxLocations%, MaxHeight%, NumDeflectors%)
  FOR J% = 1 TO MaxHeight%
    REDIM Count%(MaxLocations%)
    FOR I% = 1 TO NumStations%
      PL% = PointLocation%(I%) + 1
      IF MeanNormDefl(I%, J%, 1) > 0 THEN   'if geophone 1 is <> 0
        Count%(PL%) = Count%(PL%) + 1
        FOR K% = 1 TO NumDeflectors%
          PassSum(PL%, J%, K%) = PassSum(PL%, J%, K%) + MeanNormDefl(I%, J%, K%)
          PassSumSq(PL%, J%, K%) = PassSumSq(PL%, J%, K%) + MeanNormDefl(I%, J%, K%) ^ 2
        NEXT K%
      END IF
    NEXT I%
    FOR I% = 1 TO MaxLocations%
      FOR K% = 1 TO NumDeflectors%
        IF Count%(I%) > 0 THEN          'tests at this location
          OverallMeanNormDefl(I%, J%, K%) = PassSum(I%, J%, K%) / Count%(I%)
          IF Count%(I%) >= 3 THEN
            OverallStDevNormDefl(I%, J%, K%) = StandardDeviation(PassSum(I%, J%, K%), PassSumSq(I%, J%, K%), Count%(I%))
            OverallCVNormDefl(I%, J%, K%) = OverallStDevNormDefl(I%, J%, K%) / OverallMeanNormDefl(I%, J%, K%) * 100
          END IF
        END IF
      NEXT K%
    NEXT I%
  NEXT J%
  ERASE PassSum, PassSumSq, Count%
END SUB

SUB CalcStationStats (StatsFailed%) STATIC
' Calc average normalized deflection for each drop height at each station:
  REDIM MeanNormDefl(NumStations%, MaxHeight%, NumDeflectors% + 1)
  REDIM TPMeanNormDefl(2, MaxHeight%, NumDeflectors% + 1)

  StatsFailed% = False%

  FOR I% = 1 TO NumStations%
```

```
    REDIM Sum(NumDeflectors%)
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF Defl(I%, J%, 1) > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            IF Defl(I%, J%, K%) > 0 THEN
              Sum(K%) = Sum(K%) + Defl(I%, J%, K%) / TestLoad(I%, J%)
            ELSE
              REDIM PUText$(4)
              PUText$(1) = "Unusable deflection data found..."
              PUText$(2) = "          Station" + STR$(Station%(I%)) + " feet"
              PUText$(3) = "           Height" + STR$(Heights%(J%))
              PUText$(4) = "         Geophone" + STR$(K%)
              CALL PopupError
              StatsFailed% = True%
              GOTO ClearOut
            END IF
          NEXT K%
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            MeanNormDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
            Sum(K%) = Defl(I%, J%, K%) / TestLoad(I%, J%)
          NEXT K%
        END IF
        LastHeight% = Heights%(J%)
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            MeanNormDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
          NEXT K%
        END IF
      END IF
    NEXT J%
  NEXT I%
' Calc average normalized deflection for each drop height at each test pit:
  FOR I% = 1 TO NumTPStations%
    REDIM Sum(NumDeflectors%)
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF TPDefl(I%, J%, 1) > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            IF TPDefl(I%, J%, K%) > 0 THEN
              Sum(K%) = Sum(K%) + TPDefl(I%, J%, K%) / TPTestLoad(I%, J%)
            ELSE
              REDIM PUText$(4)
              PUText$(1) = "Unusable test pit deflection data found..."
              PUText$(2) = "          Station " + LTRIM$(STR$(TPStation%(I%))) + " feet"
              PUText$(3) = "           Height" + STR$(Heights%(J%))
              PUText$(4) = "         Geophone" + STR$(K%)
              CALL PopupError
              StatsFailed% = True%
              GOTO ClearOut
            END IF
          NEXT K%
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            TPMeanNormDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
            Sum(K%) = TPDefl(I%, J%, K%) / TPTestLoad(I%, J%)
          NEXT K%
        END IF
        LastHeight% = Heights%(J%)
```

```
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            TPMeanNormDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
          NEXT K%
        END IF
      END IF
    NEXT J%
  NEXT I%

ClearOut:
  ERASE Sum
END SUB

SUB CalcSubSectionAlpha (Recalc%) STATIC
  REDIM SubSectMean(NumSubSect%), SubSectStdDev(NumSubSect%)
  REDIM PUText$((NumSubSect% - 1) * 2 + 4), Status%(NumSubSect% - 1, 2)
  IF NOT HeightList%(2) THEN
    REDIM PUText$(2)
    PUText$(1) = "NO data present for drop height 2.  Unable to calculate statistics"
    PUText$(2) = "for these subsections.  Resetting boundaries for NO subsections..."
    CALL PopupError
    NumSubSect% = 1
    SubSectEnd%(1) = 500
    Recalc% = False%
    EXIT SUB
  END IF
  Format$ = "###.##"
  LastStation% = 0
  AnyEqual% = False%
  FOR I% = 1 TO NumSubSect%              'stats based on CORRECTED geophone 1
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(I%) + FirstStation% - 1
    SubSectSum = 0
    SubSectSumSq = 0
    FOR J% = FirstStation% TO LastStation%
      SubSectSum = SubSectSum + MeanCorrDefl(J%, 2)
      SubSectSumSq = SubSectSumSq + MeanCorrDefl(J%, 2) ^ 2
    NEXT J%
    SubSectMean(I%) = SubSectSum / StationCount%(I%)
    SubSectStdDev(I%) = SQR((SubSectSumSq - (SubSectSum ^ 2) / StationCount%(I%)) / (StationCount%(I%) - 1))
  NEXT I%
  FOR I% = 1 TO NumSubSect% - 1          'students t-test for equal means (95% confidence)
    Index% = 2 * I% - 1
    var1 = SubSectStdDev(I%) ^ 2
    var2 = SubSectStdDev(I% + 1) ^ 2
    T1 = var1 / StationCount%(I%)
    T2 = var2 / StationCount%(I% + 1)
    c1 = StationCount%(I%) - 1
    c2 = StationCount%(I% + 1) - 1
    Degrees = (T1 + T2) ^ 2 / ((T1 ^ 2 / c1) + (T2 ^ 2 / c2))
    T = (SubSectMean(I%) - SubSectMean(I% + 1)) / SQR(T1 + T2)
    aa = Degrees / 2
    bb = .5
    cc = Degrees / (Degrees + ABS(T) ^ 2)
    KeyValue = (1 - BetaI(aa, bb, cc) / 2) * 100
    IF KeyValue < ThresholdAlpha THEN
      PUText$(Index%) = "Comparing subsections" + STR$(I%) + " and" + STR$(I% + 1) + ", BELOW threshold, therefore EQUAL
means."
      AnyEqual% = True%
      Status%(I%, 1) = True%
    ELSE
      PUText$(Index%) = "Comparing subsections" + STR$(I%) + " and" + STR$(I% + 1) + ", ABOVE threshold, therefore UNEQUAL
means."
    END IF
    IF var1 > var2 THEN                  'F-Test for significantly diff variances
      FStat = var1 / var2
      df1 = c1
      df2 = c2
    ELSE
      FStat = var2 / var1
```

```
      df1 = c2
      df2 = c1
    END IF
    aa = df1 / 2
    bb = df2 / 2
    cc = df2 / (df2 + df1 * FStat)
    dd = df1 / (df1 + df2 / FStat)
    prob = Betai(bb, aa, cc) + (1 - Betai(aa, bb, dd))
    KeyValue2 = (1 - prob) * 100
    IF KeyValue2 < ThresholdAlpha THEN
      PUText$(Index% + 1) = "Comparing subsections" + STR$(I%) + " and" + STR$(I% + 1) + ", BELOW threshold, therefore EQUAL
variances."
      AnyEqual% = True%
      Status%(I%, 2) = True%
    ELSE
      PUText$(Index% + 1) = "Comparing subsections" + STR$(I%) + " and" + STR$(I% + 1) + ", ABOVE threshold, therefore
UNEQUAL variances."
    END IF
  NEXT I%
  IF AnyEqual% THEN
    PUText$((NumSubSect% - 1) * 2 + 1) = ""
    PUText$((NumSubSect% - 1) * 2 + 2) = "   At least one (1) of the comparisons shows EQUAL subsection statistics."
    PUText$((NumSubSect% - 1) * 2 + 3) = "   Do you want to re-divide this section into different subsections based"
    PUText$((NumSubSect% - 1) * 2 + 4) = "   on these results and then recalculate the comparison?"
    Recalc$ = "Y"
    CALL PopupWarning((NumSubSect% - 1) * 2 + 4, 1, Recalc$)
    Recalc$ = UCASE$(Recalc$)
    IF Recalc$ = "Y" THEN Recalc% = True% ELSE Recalc% = False%
  ELSE
    Recalc% = False%
  END IF
  ERASE SubSectMean, SubSectStdDev, PUText$
END SUB

SUB CalcSubSectionStats STATIC
' Calc subsection avg & std dev of deflection for each height and sensor:
  REDIM SubSectionMeanDefl(MaxLocations%, MaxHeight%, NumDeflectors%, NumSubSect%)
  REDIM SubSectionStDevDefl(MaxLocations%, MaxHeight%, NumDeflectors%, NumSubSect%)
  REDIM SubSectionCVDefl(MaxLocations%, MaxHeight%, NumDeflectors%, NumSubSect%)
  LastStation% = 0
  FOR L% = 1 TO NumSubSect%
    REDIM SubSectionSum(MaxLocations%, MaxHeight%, NumDeflectors%)
    REDIM SubSectionSumSq(MaxLocations%, MaxHeight%, NumDeflectors%)
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(L%) + FirstStation% - 1
    FOR J% = 1 TO MaxHeight%
      REDIM Count%(MaxLocations%)
      FOR I% = FirstStation% TO LastStation%      'for given subsection
        PL% = PointLocation%(I%) + 1
        IF Corrected% THEN                        'if data has been temp corrected
          IF MeanCorrDefl(I%, J%) > 0 THEN        'if geophone 1 is <> 0
            Count%(PL%) = Count%(PL%) + 1
            SubSectionSum(PL%, J%, 1) = SubSectionSum(PL%, J%, 1) + MeanCorrDefl(I%, J%)
            SubSectionSumSq(PL%, J%, 1) = SubSectionSumSq(PL%, J%, 1) + MeanCorrDefl(I%, J%) ^ 2
            FOR K% = 2 TO NumDeflectors%
              SubSectionSum(PL%, J%, K%) = SubSectionSum(PL%, J%, K%) + MeanNormDefl(I%, J%, K%)
              SubSectionSumSq(PL%, J%, K%) = SubSectionSumSq(PL%, J%, K%) + MeanNormDefl(I%, J%, K%) ^ 2
            NEXT K%
          END IF
        ELSE                                      'if data has NOT been temp corrected
          IF MeanNormDefl(I%, J%, 1) > 0 THEN     'if geophone 1 is <> 0
            Count%(PL%) = Count%(PL%) + 1
            FOR K% = 1 TO NumDeflectors%
              SubSectionSum(PL%, J%, K%) = SubSectionSum(PL%, J%, K%) + MeanNormDefl(I%, J%, K%)
              SubSectionSumSq(PL%, J%, K%) = SubSectionSumSq(PL%, J%, K%) + MeanNormDefl(I%, J%, K%) ^ 2
            NEXT K%
          END IF
        END IF
      NEXT I%
      FOR I% = 1 TO MaxLocations%
        IF Count%(I%) > 0 THEN                    'tests at this location
          FOR K% = 1 TO NumDeflectors%
            SubSectionMeanDefl(I%, J%, K%, L%) = SubSectionSum(I%, J%, K%) / Count%(I%)
```

```
          IF Count%(I%) >= 3 THEN
            SubSectionStDevDefl(I%, J%, K%, L%) = StandardDeviation(SubSectionSum(I%, J%, K%), SubSectionSumSq(I%, J%,
K%), Count%(I%))
            SubSectionCVDefl(I%, J%, K%, L%) = SubSectionStDevDefl(I%, J%, K%, L%) / SubSectionMeanDefl(I%, J%, K%, L%)
* 100
          END IF
        NEXT K%
      END IF
    NEXT I%
  NEXT J%
NEXT L%
ERASE SubSectionSum, SubSectionSumSq
END SUB

SUB CalcTempCorrDefl STATIC
  REDIM ESMG(NumDeflectors%), TempRadial(NumDeflectors%)
  REDIM TempCorrDefl(NumStations%, InitNumPeaks%), TPTempCorrDefl(2, InitNumPeaks%)
  StanTemp = 68                          'standard temperature
  N2 = .45                               'subgrade poisson's ratio
  P2 = 1 - N2 ^ 2
  FOR K% = 1 TO NumDeflectors%
    TempRadial(K%) = Radial(K%) * RadFactor
  NEXT K%
  E1s = 10 ^ (6.464351 - .0001454 * (StanTemp ^ 1.948)) / 1000
  FOR I% = 1 TO NumStations%
    E1f = E1s / (10 ^ (.0001454 * ((MidDepthTemp(I%) ^ 1.948) - (StanTemp ^ 1.948))))
    FOR J% = 1 TO InitNumPeaks%
      Pressure = (TestLoad(I%, J%) * LoadFactor) / (3.141593 * TLPR ^ 2)
      FOR K% = 1 TO NumDeflectors%       'deflection in inches, radial offset in inches
        IF Defl(I%, J%, K%) <> 0 THEN
          IF TempRadial(K%) < (.25 * TLPR) THEN
            ESMG(K%) = 2 * P2 * Pressure * TLPR / (Defl(I%, J%, K%) * DeflFactor)
          ELSE
            c1 = 1.1 * (LOG(TempRadial(K%) / TLPR) / LOG(10)) + 1.15
            c2 = .5 * N2 + .875
            IF c1 > c2 THEN cc = c2 ELSE cc = c1
            ESMG(K%) = (Pressure * (TLPR ^ 2) * P2 * cc) / ((Defl(I%, J%, K%) * DeflFactor) * TempRadial(K%))
          END IF
        ELSE
          ESMG(K%) = 1E+07
        END IF
      NEXT K%
      Modulus(NumLayers% + 1) = MinS!(SEG ESMG(1), NumDeflectors%) / 1000
      TempCorrDefl(I%, J%) = Defl(I%, J%, 1) * DeflectionRatio(E1s, E1f)
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    E1f = E1s / (10 ^ (.0001454 * ((TPMidDepthTemp(I%) ^ 1.948) - (StanTemp ^ 1.948))))
    FOR J% = 1 TO InitNumPeaks%
      Pressure = (TPTestLoad(I%, J%) * LoadFactor) / (3.141593 * TLPR ^ 2)
      FOR K% = 1 TO NumDeflectors%       'deflection in inches, radial offset in inches
        IF TempRadial(K%) < (.25 * TLPR) THEN
          ESMG(K%) = 2 * P2 * Pressure * TLPR / (TPDefl(I%, J%, K%) * DeflFactor)
        ELSE
          c1 = 1.1 * (LOG(TempRadial(K%) / TLPR) / LOG(10)) + 1.15
          c2 = .5 * N2 + .875
          IF c1 > c2 THEN cc = c2 ELSE cc = c1
          ESMG(K%) = (Pressure * (TLPR ^ 2) * P2 * cc) / ((TPDefl(I%, J%, K%) * DeflFactor) * TempRadial(K%))
        END IF
      NEXT K%
      Modulus(NumLayers% + 1) = MinS!(SEG ESMG(1), NumDeflectors%) / 1000
      TPTempCorrDefl(I%, J%) = TPDefl(I%, J%, 1) * DeflectionRatio(E1s, E1f)
    NEXT J%
  NEXT I%
  ERASE ESMG, TempRadial
END SUB

SUB CalcTPOverallStats STATIC
' Calc pass avg & std dev of norm deflection for each height and sensor:
  REDIM PassSum(MaxHeight%, NumDeflectors%)
  REDIM PassSumSq(MaxHeight%, NumDeflectors%)
  REDIM TPOverallMeanNormDefl(MaxHeight%, NumDeflectors%)
  REDIM TPOverallStDevNormDefl(MaxHeight%, NumDeflectors%)
```

```
  REDIM TPOverallCVNormDefl(MaxHeight%, NumDeflectors%)
  FOR J% = 1 TO MaxHeight%
    ACount% = 0
    FOR I% = 1 TO NumTPStations%
      IF TPMeanNormDefl(I%, J%, 1) > 0 THEN   'if geophone 1 is <> 0
        ACount% = ACount% + 1
        FOR K% = 1 TO NumDeflectors%
          PassSum(J%, K%) = PassSum(J%, K%) + TPMeanNormDefl(I%, J%, K%)
          PassSumSq(J%, K%) = PassSumSq(J%, K%) + TPMeanNormDefl(I%, J%, K%) ^ 2
        NEXT K%
      END IF
    NEXT I%
    FOR K% = 1 TO NumDeflectors%
      IF ACount% > 0 THEN          'tests at this location
        TPOverallMeanNormDefl(J%, K%) = PassSum(J%, K%) / ACount%
        IF ACount% >= 2 THEN
          TPOverallStDevNormDefl(J%, K%) = StandardDeviation(PassSum(J%, K%), PassSumSq(J%, K%), ACount%)
          TPOverallCVNormDefl(J%, K%) = TPOverallStDevNormDefl(J%, K%) / TPOverallMeanNormDefl(J%, K%) * 100
        END IF
      END IF
    NEXT K%
  NEXT J%
  ERASE PassSum, PassSumSq
END SUB

FUNCTION DeflectionRatio (E1s, E1f) STATIC
  REDIM AlphaS(MaxLayers%, MaxLayers%), AlphaF(MaxLayers%, MaxLayers%)
  REDIM TransThickS(MaxLayers%, MaxLayers%), TransThickF(MaxLayers%, MaxLayers%)
' PRINT USING "####.#  "; Modulus(NumLayers% + 1); E1s; E1f
  CompressionS = 0
  CompressionF = 0
  FOR L% = 1 TO NumLayers% + 1
    FOR M% = L% TO NumLayers% + 1
      IF L% = 1 THEN
        IF M% = 1 THEN
          AlphaS(L%, M%) = 1
          AlphaF(L%, M%) = 1
          TransThickS(L%, M%) = Thickness(L%)
          TransThickF(L%, M%) = Thickness(L%)
        ELSE
          AlphaS(L%, M%) = 1 - (Log10(E1s / Modulus(M%)) / (7.5 * Thickness(L%) ^ .2))
          AlphaF(L%, M%) = 1 - (Log10(E1f / Modulus(M%)) / (7.5 * Thickness(L%) ^ .2))
          TransThickS(L%, M%) = AlphaS(L%, M%) * Thickness(L%) * ((E1s / Modulus(M%)) ^ .333333)
          TransThickF(L%, M%) = AlphaF(L%, M%) * Thickness(L%) * ((E1f / Modulus(M%)) ^ .333333)
        END IF
      ELSEIF L% = NumLayers% + 1 THEN
        'continue, subgrade layer never transformed
      ELSE
        AlphaS(L%, M%) = 1 - (Log10(Modulus(L%) / Modulus(M%)) / (7.5 * Thickness(L%) ^ .2))
        AlphaF(L%, M%) = 1 - (Log10(Modulus(L%) / Modulus(M%)) / (7.5 * Thickness(L%) ^ .2))
        TransThickS(L%, M%) = AlphaS(L%, M%) * Thickness(L%) * ((Modulus(L%) / Modulus(M%)) ^ .333333)
        TransThickF(L%, M%) = AlphaF(L%, M%) * Thickness(L%) * ((Modulus(L%) / Modulus(M%)) ^ .333333)
      END IF
'     PRINT USING "##  ##  #.###  #.###  ###.##  ###.##"; L%; M%; AlphaS(L%, M%); AlphaF(L%, M%); TransThickS(L%, M%);
TransThickF(L%, M%)
    NEXT M%
    TopSumS = 0: TopSumF = 0
    BottomSumS = 0: BottomSumF = 0
    IF L% = 1 THEN
      TopSumS = 0: TopSumF = 0
      BottomSumS = Thickness(L%)
      BottomSumF = Thickness(L%)
    ELSE
      FOR M% = 1 TO L% - 1
        TopSumS = TopSumS + TransThickS(M%, L%)
        TopSumF = TopSumF + TransThickF(M%, L%)
      NEXT M%
      IF L% <> NumLayers% + 1 THEN
        BottomSumS = TopSumS + Thickness(L%)
        BottomSumF = TopSumF + Thickness(L%)
      END IF
    END IF
    IF L% = 1 THEN
```

```
      FbsTop = 1
      FbfTop = 1
    ELSE
      FbsTop = 1 / (SQR(1 + (TopSumS / TLPR) ^ 2))
      FbfTop = 1 / (SQR(1 + (TopSumF / TLPR) ^ 2))
    END IF
    IF L% <> NumLayers% + 1 THEN
      FbsBottom = 1 / (SQR(1 + (BottomSumS / TLPR) ^ 2))
      FbfBottom = 1 / (SQR(1 + (BottomSumF / TLPR) ^ 2))
    ELSE
      FbsBottom = 0
      FbfBottom = 0
    END IF
    ModRatio = Modulus(NumLayers% + 1) / Modulus(L%)
    CompressionS = CompressionS + (FbsTop - FbsBottom) * ModRatio
    CompressionF = CompressionF + (FbfTop - FbfBottom) * ModRatio
'   PRINT USING "#.####  #.####  #.####  #.####"; FbsTop; FbsBottom; FbfTop; FbfBottom
  NEXT L%
' PRINT CompressionS / CompressionF
  DeflectionRatio = CompressionS / CompressionF
END FUNCTION

SUB GetCorrectedDeflectionParameters STATIC
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN EXIT FOR
  NEXT PlotHeight%
  FOR PlotLocation% = 1 TO MaxLocations%
    IF LocationList%(PlotLocation% + 1) THEN EXIT FOR
  NEXT PlotLocation%
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%))
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = MeanCorrDefl(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = TPMeanCorrDefl(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  PlotYMax = PlotYMax * 1000!
  T1 = INT(LOG(PlotYMax) / LOG(10))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  DoPlot% = True%
  DO
    IF DoPlot% THEN CALL PlotCorrDeflCurves(PlotLocation%, PlotHeight%, PlotYMax)
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    DoPlot% = True%
    SELECT CASE KCode%
      CASE 72               'up arrow, increase height by 1
        DO
          PlotHeight% = PlotHeight% + 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
          IF PlotHeight% > MaxHeight% THEN PlotHeight% = 0
        LOOP
      CASE 80               'down arrow, decrease height by 1
        DO
          PlotHeight% = PlotHeight% - 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
```

```
        IF PlotHeight% < 1 THEN PlotHeight% = MaxHeight% + 1
      LOOP
    CASE 73                 'PgUp, decrease location by 1
      LastLoc% = PlotLocation%
      DO
        PlotLocation% = PlotLocation% - 1
        IF LocationList%(PlotLocation% + 1) AND PlotLocation% <> 0 THEN EXIT DO
        IF PlotLocation% < 1 THEN PlotLocation% = MaxLocations% + 1
      LOOP
      IF PlotLocation% = LastLoc% THEN DoPlot% = False%
    CASE 81                 'PgDn, increase location by 1
      LastLoc% = PlotLocation%
      DO
        PlotLocation% = PlotLocation% + 1
        IF LocationList%(PlotLocation% + 1) THEN EXIT DO
        IF PlotLocation% > MaxLocations% THEN PlotLocation% = 0
      LOOP
      IF PlotLocation% = LastLoc% THEN DoPlot% = False%
    CASE 60                 'F2:screen dump
      CALL ScrnDump(ScrnDumpRes$, 1, 0)
      LPRINT CHR$(12);
      DoPlot% = False%
    CASE 68                 'F10:ExitPlots
      EXIT SUB
    CASE ELSE
      DoPlot% = False%
  END SELECT
LOOP
END SUB

SUB GetOutlierCurveParameters (ExitCode%) STATIC
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN EXIT FOR
  NEXT PlotHeight%
  FOR PlotLocation% = 1 TO MaxLocations%
    IF LocationList%(PlotLocation% + 1) THEN EXIT FOR
  NEXT PlotLocation%
  PlotDeflector% = 1
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%) * NumDeflectors%)
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        FOR K% = 1 TO NumDeflectors%
          Count% = Count% + 1
          Y(Count%) = Deviation(I%, J%, K%)
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        FOR K% = 1 TO NumDeflectors%
          Count% = Count% + 1
          Y(Count%) = TPDeviation(I%, J%, K%)
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  PlotYMin = MinS!(SEG Y(1), Count%)
  ERASE Y
  IF ABS(PlotYMin) > PlotYMax THEN PlotYMax = ABS(PlotYMin)
  T1 = INT(LOG(PlotYMax) / LOG(10))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  IF PlotYMax < 3 THEN PlotYMax = 3
  PlotYMin = -PlotYMax
  DoPlot% = True%
  DO
    IF DoPlot% THEN CALL PlotDeviationCurve(PlotLocation%, PlotHeight%, PlotDeflector%, PlotYMin, PlotYMax)
```

```
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    DoPlot% = True%
    SELECT CASE KCode%
      CASE 75               'left arrow, decrease deflector by 1
        PlotDeflector% = PlotDeflector% - 1
        IF PlotDeflector% < 1 THEN PlotDeflector% = MaxDeflectors%
      CASE 77               'right arrow, increase deflector by 1
        PlotDeflector% = PlotDeflector% + 1
        IF PlotDeflector% > MaxDeflectors% THEN PlotDeflector% = 1
      CASE 72               'up arrow, increase height by 1
        DO
          PlotHeight% = PlotHeight% + 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
          IF PlotHeight% > MaxHeight% THEN PlotHeight% = 0
        LOOP
      CASE 80               'down arrow, decrease height by 1
        DO
          PlotHeight% = PlotHeight% - 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
          IF PlotHeight% < 1 THEN PlotHeight% = MaxHeight% + 1
        LOOP
      CASE 73               'PgUp, decrease location by 1
        LastLoc% = PlotLocation%
        DO
          PlotLocation% = PlotLocation% - 1
          IF LocationList%(PlotLocation% + 1) AND PlotLocation% <> 0 THEN EXIT DO
          IF PlotLocation% < 1 THEN PlotLocation% = MaxLocations% + 1
        LOOP
        IF PlotLocation% = LastLoc% THEN DoPlot% = False%
      CASE 81               'PgDn, increase location by 1
        LastLoc% = PlotLocation%
        DO
          PlotLocation% = PlotLocation% + 1
          IF LocationList%(PlotLocation% + 1) THEN EXIT DO
          IF PlotLocation% > MaxLocations% THEN PlotLocation% = 0
        LOOP
        IF PlotLocation% = LastLoc% THEN DoPlot% = False%
      CASE 60               'F2:screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
        DoPlot% = False%
      CASE 68               'F10:ExitPlots
        EXIT SUB
      CASE ELSE
        DoPlot% = False%
    END SELECT
  LOOP
END SUB

SUB GetUniformityCurveParameters STATIC
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN EXIT FOR
  NEXT PlotHeight%
  FOR PlotLocation% = 1 TO MaxLocations%
    IF LocationList%(PlotLocation% + 1) THEN EXIT FOR
  NEXT PlotLocation%
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%) * NumDeflectors%)
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        FOR K% = 1 TO NumDeflectors%
          Count% = Count% + 1
          Y(Count%) = MeanNormDefl(I%, J%, K%)
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
```

```
      IF HeightList%(J%) THEN
        FOR K% = 1 TO NumDeflectors%
          Count% = Count% + 1
          Y(Count%) = TPMeanNormDefl(I%, J%, K%)
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  PlotYMax = PlotYMax * 1000!
  T1 = INT(LOG(PlotYMax) / LOG(10))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  DoPlot% = True%
  DO
    IF DoPlot% THEN CALL PlotNormDeflCurves(PlotLocation%, PlotHeight%, PlotYMax)
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    DoPlot% = True%
    SELECT CASE KCode%
      CASE 72                'up arrow, increase height by 1
        DO
          PlotHeight% = PlotHeight% + 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
          IF PlotHeight% > MaxHeight% THEN PlotHeight% = 0
        LOOP
      CASE 80                'down arrow, decrease height by 1
        DO
          PlotHeight% = PlotHeight% - 1
          IF HeightList%(PlotHeight%) THEN EXIT DO
          IF PlotHeight% < 1 THEN PlotHeight% = MaxHeight% + 1
        LOOP
      CASE 73                'PgUp, decrease location by 1
        LastLoc% = PlotLocation%
        DO
          PlotLocation% = PlotLocation% - 1
          IF LocationList%(PlotLocation% + 1) AND PlotLocation% <> 0 THEN EXIT DO
          IF PlotLocation% < 1 THEN PlotLocation% = MaxLocations% + 1
        LOOP
        IF PlotLocation% = LastLoc% THEN DoPlot% = False%
      CASE 81                'PgDn, increase location by 1
        LastLoc% = PlotLocation%
        DO
          PlotLocation% = PlotLocation% + 1
          IF LocationList%(PlotLocation% + 1) THEN EXIT DO
          IF PlotLocation% > MaxLocations% THEN PlotLocation% = 0
        LOOP
        IF PlotLocation% = LastLoc% THEN DoPlot% = False%
      CASE 60                'F2:screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
        DoPlot% = False%
      CASE 68                'F10:ExitPlots
        EXIT SUB
      CASE ELSE
        DoPlot% = False%
    END SELECT
  LOOP
END SUB

SUB PlotCorrDeflCurves (PlotLocation%, PlotHeight%, PlotYMax) STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 25: PRINT "Corrected Deflection Data for Section: "; PS$
  LOCATE 26, 1: PRINT "Corrected"
  LOCATE 27, 1: PRINT "Normalized"
  LOCATE 28, 1: PRINT "Deflection"
  FOR Value% = -100 TO 600 STEP 100
```

```
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F2:ScrnDump  F10:Exit  " + CHR$(25) + CHR$(24) + ":Prv/Nxt Ht" + "  PgUp/PgDn:Prv/Nxt Loc"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3              'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "#.###^^^^"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 18: PRINT "Location"; PlotLocation%; "    Drop Height"; PlotHeight%; "    Sensors";
  FOR Pass% = 1 TO 2
    SELECT CASE Pass%
      CASE 1
        PlotDeflector% = 1: LineColor% = 5
      CASE 2
        PlotDeflector% = 7: LineColor% = 11
    END SELECT
    Started% = False%
    FOR I% = 1 TO NumStations%
      IF PointLocation%(I%) = PlotLocation% THEN
        PX = X0 + (Station%(I%) - PlotXMin) / XScale
        IF PlotDeflector% = 1 THEN
          PY = Y0 - ((MeanCorrDefl(I%, PlotHeight%) * 1000!) - PlotYMin) / YScale
        ELSE
          PY = Y0 - ((MeanNormDefl(I%, PlotHeight%, PlotDeflector%) * 1000!) - PlotYMin) / YScale
        END IF
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      END IF
    NEXT I%
    FOR I% = 1 TO NumTPStations%
      PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
      IF PlotDeflector% = 1 THEN
        PY = Y0 - ((TPMeanCorrDefl(I%, PlotHeight%) * 1000!) - PlotYMin) / YScale
      ELSE
        PY = Y0 - ((TPMeanNormDefl(I%, PlotHeight%, PlotDeflector%) * 1000!) - PlotYMin) / YScale
      END IF
      CIRCLE (PX, PY), 3, LineColor%
      PAINT (PX, PY), LineColor%, LineColor%
    NEXT I%
    COLOR LineColor%
    PRINT QPRTrim$(STR$(PlotDeflector%));
    IF Pass% <> 2 THEN COLOR 15: PRINT ",";
  NEXT Pass%
END SUB

SUB PlotDeviationCurve (PlotLocation%, PlotHeight%, PlotDeflector%, PlotYMin, PlotYMax) STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 25: PRINT "Deflection Deviation Data for Section: "; PS$
  LOCATE 27, 2: PRINT "Standard"
  LOCATE 28, 1: PRINT "Deviations"
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F2:ScrnDump  F10:Exit  " + CHR$(25) + CHR$(24) + ":Prv/Nxt Ht  " + CHR$(27) + CHR$(26) + ":Prv/Nxt Defl
PgUp/PgDn:Prv/Nxt Loc"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  LY = 400: Y0 = 420: X0 = XMin
```

```
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = PlotYMin + 1 TO PlotYMax - 1          'tic marks
    PM = YO - (M - PlotYMin) / YScale
    LINE (XO, PM)-(XO + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "###.#"
  COLOR 15
  LOCATE 53, 11: PRINT USING FA$; PlotYMin;
  LOCATE 28, 11: PRINT USING FA$; 0
  LOCATE 3, 11: PRINT USING FA$; PlotYMax
  LineColor% = 11
  LOCATE 58, 18: PRINT "Location"; PlotLocation%; "    Drop Height"; PlotHeight%; "    Sensor"; PlotDeflector%
  Started% = False%
  FOR I% = 1 TO NumStations%
    IF PointLocation%(I%) = PlotLocation% THEN
      PX = XO + (Station%(I%) - PlotXMin) / XScale
      PY = YO - ((Deviation(I%, PlotHeight%, PlotDeflector%)) - PlotYMin) / YScale
      IF Started% THEN LINE -(PX, PY), LineColor%
      CIRCLE (PX, PY), 3, LineColor%
      PAINT (PX + 1, PY), LineColor%, LineColor%
      Started% = True%
    END IF
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    PX = XO + (TPStation%(I%) - PlotXMin) / XScale
    PY = YO - ((TPDeviation(I%, PlotHeight%, PlotDeflector%)) - PlotYMin) / YScale
    CIRCLE (PX, PY), 3, 10
    PAINT (PX, PY), 10, 10
  NEXT I%
END SUB

SUB PlotNormDeflCurves (PlotLocation%, PlotHeight%, PlotYMax) STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 30: PRINT "Deflection Data for Section: "; PS$
  LOCATE 27, 1: PRINT "Normalized"
  LOCATE 28, 1: PRINT "Deflection"
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F2:ScrnDump  F10:Exit  " + CHR$(25) + CHR$(24) + ":Prv/Nxt Ht" + "  PgUp/PgDn:Prv/Nxt Loc"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  PlotYMin = 0
  LY = 400: YO = 420: XO = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = YO - M * LY / 4
    LINE (XO, PM)-(XO + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "#.###^^^^"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 18: PRINT "Location"; PlotLocation%; "    Drop Height"; PlotHeight%; "    Sensors";
  FOR Pass% = 1 TO NumDeflectors%
    SELECT CASE Pass%
      CASE 1
        PlotDeflector% = 1: LineColor% = 5
      CASE 2
        PlotDeflector% = 2: LineColor% = 9
      CASE 3
        PlotDeflector% = 3: LineColor% = 10
      CASE 4
        PlotDeflector% = 4: LineColor% = 1
      CASE 5
        PlotDeflector% = 5: LineColor% = 2
      CASE 6
        PlotDeflector% = 6: LineColor% = 3
      CASE 7
```

```
        PlotDeflector% = 7: LineColor% = 11
    END SELECT
    Started% = False%
    FOR I% = 1 TO NumStations%
      IF PointLocation%(I%) = PlotLocation% THEN
        PX = X0 + (Station%(I%) - PlotXMin) / XScale
        PY = Y0 - ((MeanNormDefl(I%, PlotHeight%, PlotDeflector%) * 1000!) - PlotYMin) / YScale
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      END IF
    NEXT I%
    FOR I% = 1 TO NumTPStations%
      PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
      PY = Y0 - ((TPMeanNormDefl(I%, PlotHeight%, PlotDeflector%) * 1000!) - PlotYMin) / YScale
      CIRCLE (PX, PY), 3, LineColor%
      PAINT (PX, PY), LineColor%, LineColor%
    NEXT I%
    COLOR LineColor%
    PRINT QPRTrim$(STR$(PlotDeflector%));
    IF Pass% <> NumDeflectors% THEN COLOR 15: PRINT ",";
  NEXT Pass%
END SUB

SUB PlotSetup (XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale) STATIC
  SCREEN 12: WIDTH 80, 60: COLOR 0: CLS
  XMin = 130: XMax = 630: YMin = 20: YMax = 420
  LINE (XMin, YMin)-(XMax, YMax), 6, B
  LX = 500
  PlotXMin = -100: PlotXMax = 600
  XScale = ABS(PlotXMax - PlotXMin) / LX
  FOR M = 0 TO 500 STEP 100              'station offsets
    PM = XMin + (M - PlotXMin) / XScale
    LINE (PM, YMin)-(PM, YMax), 7, , &HCCCC
  NEXT M
  IF NumSubSect% > 1 THEN                'plot subsection boundaries, if known
    FOR I% = 1 TO NumSubSect% - 1
      PM = XMin + (SubSectEnd%(I%) - PlotXMin) / XScale
      LINE (PM, YMin)-(PM, YMax), 14
    NEXT I%
  END IF
END SUB

SUB RedimArrays STATIC
  REDIM Radial(MaxDeflectors%), LocationList%(MaxLocations%), HeightList%(MaxHeights%)
  REDIM PointLocation%(MaxNumStations%), Heights%(MaxNumPeaks%)
  REDIM SubSectEnd%(5), StationCount%(5)
  REDIM Thickness(MaxLayers%), Modulus(MaxLayers%), MatlCode%(MaxLayers%)
  REDIM PoissonRatio(MaxLayers%), ALo(MaxLayers%), AHi(MaxLayers%), SelCode%(MaxLayers%)
  REDIM Station%(MaxNumStations%), TPStation%(2)
  REDIM MidDepthTemp(MaxNumStations%), TPMidDepthTemp(2)
  REDIM TestLoad(MaxNumStations%, MaxNumPeaks%), TPTestLoad(2, MaxNumPeaks%)
  REDIM Defl(MaxNumStations%, MaxNumPeaks%, MaxDeflectors%), TPDefl(2, MaxNumPeaks%, MaxDeflectors%)
  REDIM FWDMinutes%(MaxNumStations%), TPFWDMinutes%(2), MeasTime$(10, 2), NumTimes%(2)
  REDIM TimeTemp(MaxTimes%, 2, 3), MeasMinutes%(10, 2)
END SUB

SUB ResetVariables STATIC
  NumStations% = 0
  NumTPStations% = 0
  NumSubSect% = 1
  NumLayers% = 1
  SetMaterials% = False%
  Corrected% = False%
  SetSubSections% = False%
  SummaryOfResults% = False%
  SummaryOfResults2% = False%
END SUB

SUB SetMaterialInfo STATIC
  REDIM MaterialList$(21), ModulusList(21), PoissonList(21), AListLo(21), AListHi(21)
  MaterialList$(1) = "302  Uncrushed Gravel          ":  ModulusList(1) = 20:    PoissonList(1) = .4:   AListLo(1) = .07:
AListHi(1) = .17
```

```
  MaterialList$(2) = "303  Crushed Stone            ":  ModulusList(2) = 45:    PoissonList(2) = .4:   AListLo(2) = .11:
AListHi(2) = .21
  MaterialList$(3) = "304  Crushed Gravel           ":  ModulusList(3) = 30:    PoissonList(3) = .4:   AListLo(3) = .09:
AListHi(3) = .18
  MaterialList$(4) = "305  Crushed Slag             ":  ModulusList(4) = 50:    PoissonList(4) = .4:   AListLo(4) = .12:
AListHi(4) = .22
  MaterialList$(5) = "306  Sand                     ":  ModulusList(5) = 10:    PoissonList(5) = .4:   AListLo(5) = .05:
AListHi(5) = .15
  MaterialList$(6) = "307  Fine Soil-Agg. Mixture   ":  ModulusList(6) = 15:    PoissonList(6) = .4:   AListLo(6) = .06:
AListHi(6) = .16
  MaterialList$(7) = "308  Coarse Soil-Agg. Mixture ":  ModulusList(7) = 20:    PoissonList(6) = .4:   AListLo(7) = .07:
AListHi(7) = .17
  MaterialList$(8) = "320  Sand Asphalt             ":  ModulusList(8) = 200:   PoissonList(6) = .4:   AListLo(8) = .1:
AListHi(8) = .3
  MaterialList$(9) = "321  Asphalt Treated Mixture  ":  ModulusList(9) = 300:   PoissonList(9) = .35:  AListLo(9) = .15:
AListHi(9) = .35
  MaterialList$(10) = "331  Cement Aggregate Mixture": ModulusList(10) = 750:  PoissonList(10) = .3:  AListLo(10) = .25:
AListHi(10) = .45
  MaterialList$(11) = "332  Econocrete              ": ModulusList(11) = 1500: PoissonList(11) = .25: AListLo(11) = .4:
AListHi(11) = .6
  MaterialList$(12) = "333  Cement Treated Soil     ": ModulusList(12) = 100:  PoissonList(12) = .35: AListLo(12) = .1:
AListHi(12) = .25
  MaterialList$(13) = "334  Lean Concrete           ": ModulusList(13) = 1500: PoissonList(13) = .25: AListLo(13) = .4:
AListHi(13) = .6
  MaterialList$(14) = "336  Sand-Shell Mixture      ": ModulusList(14) = 75:   PoissonList(14) = .4:  AListLo(14) = .15:
AListHi(14) = .25
  MaterialList$(15) = "337  Limerock, Caliche       ": ModulusList(15) = 200:  PoissonList(15) = .35: AListLo(15) = .15:
AListHi(15) = .3
  MaterialList$(16) = "338  Lime Treated Soil       ": ModulusList(16) = 75:   PoissonList(16) = .35: AListLo(16) = .1:
AListHi(16) = .25
  MaterialList$(17) = "339  Soil Cement             ": ModulusList(17) = 200:  PoissonList(17) = .35: AListLo(17) = .15:
AListHi(17) = .3
  MaterialList$(18) = "340  Pozzolanic-Agg. Mixture ": ModulusList(18) = 500:  PoissonList(18) = .35: AListLo(18) = .2:
AListHi(18) = .4
  MaterialList$(19) = "341  Cracked & Seated PCC    ": ModulusList(19) = 1000: PoissonList(19) = .25: AListLo(19) = .35:
AListHi(19) = .45
  MaterialList$(20) = "700  Asphaltic Concrete      ": ModulusList(20) = 450:  PoissonList(20) = .35: AListLo(20) = .35:
AListHi(20) = .45
  MaterialList$(21) = "730  Portland Cement Concrete": ModulusList(21) = 5000: PoissonList(21) = .15: AListLo(21) = .6:
AListHi(21) = .8
END SUB

FUNCTION StandardDeviation (SSum, SSum2, ItemCount%) STATIC
  TempCalc = (SSum2 - (SSum ^ 2) / ItemCount%) / (ItemCount% - 1)
  IF TempCalc > 0 THEN
    StandardDeviation = SQR(TempCalc)
  ELSE
    StandardDeviation = 0
  END IF
END FUNCTION
```

```
DECLARE SUB CalcCorrStationStats ()
DECLARE SUB CalcMidDepthTemps ()
DECLARE SUB CalcOutliers ()
DECLARE SUB CalcSubSectionAlpha (Recalc%)
DECLARE SUB CalcSubSectionStats ()
DECLARE SUB CalcTempCorrDefl ()
DECLARE SUB GetCorrectedDeflectionParameters ()
DECLARE SUB GetOutlierCurveParameters (ExitCode%)
DECLARE SUB GetUniformityCurveParameters ()
DECLARE SUB ReadPeaks ()
DECLARE SUB ReadNextLine (DataType%)
DECLARE SUB ReviewMenu ()
DECLARE SUB GetRunningComments ()
DECLARE SUB GetMaterialProps ()
DECLARE SUB GetSubsectionBoundaries (ExitCode%)
DECLARE SUB CheckHeader ()
DECLARE SUB GetTemperatureGradientsControl ()
DECLARE SUB GetTemperatureGradientsPage1 (ExitCode%)
DECLARE SUB GetTemperatureGradientsPage2 (J%, ExitCode%)
DECLARE SUB WriteStructuralStats ()
DECLARE SUB ShowTPRigidStats ()
DECLARE SUB ShowTPFlexStats ()
DECLARE SUB CalcFlexStats ()
DECLARE SUB PlotEsgCurves ()
DECLARE SUB PlotFlexSNCurves ()
DECLARE SUB ShowOverallFlexStats ()
DECLARE SUB ShowOverallRigStats ()
DECLARE SUB CompositeModulusParameters ()
DECLARE SUB PlotVolKCurves ()
DECLARE SUB PlotRigThickCurves ()
DECLARE SUB CalcRigidStats ()
DECLARE SUB Quit ()
DECLARE SUB WriteRunningComments ()
DECLARE SUB WriteOutlierMsgs ()
DECLARE SUB ShowOverallStats ()
DECLARE SUB ShowSubSectionStats ()
DECLARE SUB WriteSubsectionStats ()
DECLARE SUB WriteOverallStatistics ()
DECLARE SUB CalcStationsInSubsection ()
DECLARE SUB ChevronInputs ()
DECLARE SUB CalcRigidThickness ()
DECLARE SUB CalcCompositeModulus ()
DECLARE SUB CalcVolumetricK ()
DECLARE SUB ChevronAnalysis (AnalysisFailed%)
DECLARE FUNCTION QPROUND$ (Number!, Places%)      'do not erase
DECLARE FUNCTION BetaI! (A!, B!, X!)              'do not erase
DECLARE FUNCTION BetaCF! (A!, B!, X!)             'do not erase
DECLARE FUNCTION GammaLN! (XX!)                   'do not erase
'$INCLUDE: 'declare.inc'
'$INCLUDE: 'fdeclar1.inc'
'$INCLUDE: 'fdeclar2.inc'
'$INCLUDE: 'cmnblank.inc'
'$INCLUDE: 'fwdcheck.inc'

CONST True% = -1, False% = 0, MaxNumStations% = 50, MaxLocations% = 6
CONST MaxDeflectors% = 7, MaxNumPeaks% = 16, MaxHeights% = 4, MaxLayers% = 10
CONST MinPointsInSubSection% = 4, ThresholdAlpha = 95, MaxTimes% = 10

FUNCTION BetaCF (A, B, X) STATIC
  ITMAX% = 100
  EPS = .0000003
  AM = 1!
  BM = 1!
  AZ = 1!
  QAB = A + B
  QAP = A + 1!
  QAM = A - 1!
  BZ = 1! - QAB * X / QAP
  FOR M% = 1 TO ITMAX%
    EM = M%
    TEM = EM + EM
    D = EM * (B - M) * X / ((QAM + TEM) * (A + TEM))
    AP = AZ + D * AM
```

```
    BP = BZ + D * BM
    D = -(A + EM) * (QAB + EM) * X / ((A + TEM) * (QAP + TEM))
    APP = AP + D * AZ
    BPP = BP + D * BZ
    AOLD = AZ
    AM = AP / BPP
    BM = BP / BPP
    AZ = APP / BPP
    BZ = 1!
    IF ABS(AZ - AOLD) < (EPS * ABS(AZ)) THEN
      BetaCF = AZ
      EXIT FUNCTION
    END IF
  NEXT M%
  PRINT "A or B too big, or ITMAX too small"
  STOP
END FUNCTION

FUNCTION Betai (A, B, X) STATIC
  IF X < 0! OR X > 1! THEN
    PRINT "bad argument X in BETAI"
    STOP
  END IF
  IF X = 0! OR X = 1! THEN
    BT = 0!
  ELSE
    BT = EXP(GammaLN(A + B) - GammaLN(A) - GammaLN(B) + A * LOG(X) + B * LOG(1! - X))
  END IF
  IF X < ((A + 1!) / (A + B + 2!)) THEN
    Betai = BT * BetaCF(A, B, X) / A
  ELSE
    Betai = 1! - BT * BetaCF(B, A, 1! - X) / B
  END IF
END FUNCTION

SUB CheckHeader STATIC
  CALL ReadNextLine(DataType%)
  SELECT CASE LineCounter&
    CASE 1
      FileWidth% = VAL(MID$(LineData$, 2, 4))
      IF FileWidth% = 32 THEN
        English% = False%
      ELSE
        English% = True%
      END IF
      Edition% = VAL(MID$(LineData$, 31, 2))
      IF Edition% <> 10 AND Edition% <> 20 THEN   'not valid FWD data file
        COLOR 7, 0, 0
        CLS
        PRINT "*  Not a valid Dynatest FWD data file..."
        END
      END IF
    CASE 2
      NumDeflectors% = VAL(LEFT$(LineData$, 1))
    CASE 3                      'use only metric spacings, converted to English
      LPR = VAL(MID$(LineData$, 1, 4)) / 25.4
      LPR = VAL(QPROUND$(LPR, 1))
      Radial(1) = VAL(MID$(LineData$, 5, 4)) / 25.4
      Radial(1) = VAL(QPROUND$(Radial(1), 1))
      Radial(2) = VAL(MID$(LineData$, 9, 4)) / 25.4
      Radial(2) = VAL(QPROUND$(Radial(2), 1))
      Radial(3) = VAL(MID$(LineData$, 13, 4)) / 25.4
      Radial(3) = VAL(QPROUND$(Radial(3), 1))
      Radial(4) = VAL(MID$(LineData$, 17, 4)) / 25.4
      Radial(4) = VAL(QPROUND$(Radial(4), 1))
      Radial(5) = VAL(MID$(LineData$, 21, 4)) / 25.4
      Radial(5) = VAL(QPROUND$(Radial(5), 1))
      Radial(6) = VAL(MID$(LineData$, 25, 4)) / 25.4
      Radial(6) = VAL(QPROUND$(Radial(6), 1))
      Radial(7) = VAL(MID$(LineData$, 29, 4)) / 25.4
      Radial(7) = VAL(QPROUND$(Radial(7), 1))
    CASE 4 TO 29, 32, 34 TO 36
      'do nothing
```

```
    CASE 30                             'active sequence drops
      Posit% = INSTR(LineData$, ".")
      ActiveDrops% = Posit% - 1
    CASE 31                             'sequence heights
      Heights$ = LineData$
    CASE 33                             'peaks stored
      CheckText$ = LEFT$(LineData$, ActiveDrops%)
      InitNumPeaks% = InCount2%(CheckText$, "**")
      DropNum% = 0
      FOR I% = 1 TO ActiveDrops%        'height for each drop in sequence
        StorePeak$ = MID$(CheckText$, I%, 1)
        IF StorePeak$ = "*" THEN
          HeightVal% = VAL(MID$(Heights$, I%, 1))
          IF HeightVal% <> 0 THEN
            DropNum% = DropNum% + 1
            Heights%(DropNum%) = HeightVal%
            HeightList%(HeightVal%) = True%
          END IF
        END IF
      NEXT I%
      MaxHeight% = MaxI%(SEG Heights%(1), DropNum%)
  END SELECT
END SUB

SUB CheckHeader2 (ExitCode%) STATIC
  CALL ReadNextLine(DataType%)
  SELECT CASE LineCounter&
    CASE 1
      FileWidth2% = VAL(MID$(LineData$, 2, 4))
      IF FileWidth2% = 32 THEN
        English2% = False%
      ELSE
        English2% = True%
      END IF
      IF English2% <> English% THEN
        REDIM PUText$(1)
        PUText$(1) = "Test pit data has different system of units...  Exiting Test Pit analysis"
        CALL PopupError
        ExitCode% = 999
        EXIT SUB
      END IF
      Edition2% = VAL(MID$(LineData$, 31, 2))
      IF Edition2% <> 10 AND Edition2% <> 20 THEN 'not valid FWD data file
        REDIM PUText$(1)
        PUText$(1) = "Test pit data is NOT a valid Dynatest data file...  Exiting Test Pit analysis"
        CALL PopupError
        ExitCode% = 999
        EXIT SUB
      END IF
    CASE 2
      NumDeflectors2% = VAL(LEFT$(LineData$, 1))
      IF NumDeflectors2% <> NumDeflectors% THEN
        REDIM PUText$(2)
        PUText$(1) = "Test pit data has different number of deflectors..."
        PUText$(2) = "            Skipping Test Pit analysis"
        CALL PopupError
        ExitCode% = 999
        EXIT SUB
      END IF
    CASE 3
      REDIM Radial2(NumDeflectors%)
      LPR = VAL(MID$(LineData$, 1, 4)) / 25.4
      LPR = VAL(QPROUND$(LPR, 1))
      Rad = VAL(MID$(LineData$, 5, 4)) / 25.4
      Radial2(1) = VAL(QPROUND$(Rad, 1))
      Rad = VAL(MID$(LineData$, 9, 4)) / 25.4
      Radial2(2) = VAL(QPROUND$(Rad, 1))
      Rad = VAL(MID$(LineData$, 13, 4)) / 25.4
      Radial2(3) = VAL(QPROUND$(Rad, 1))
      Rad = VAL(MID$(LineData$, 17, 4)) / 25.4
      Radial2(4) = VAL(QPROUND$(Rad, 1))
      Rad = VAL(MID$(LineData$, 21, 4)) / 25.4
      Radial2(5) = VAL(QPROUND$(Rad, 1))
```

```
      Rad = VAL(MID$(LineData$, 25, 4)) / 25.4
      Radial2(6) = VAL(QPROUND$(Rad, 1))
      Rad = VAL(MID$(LineData$, 29, 4)) / 25.4
      Radial2(7) = VAL(QPROUND$(Rad, 1))
      FOR I% = 1 TO NumDeflectors%
        IF ABS(Radial(I%) - Radial2(I%)) > .01 THEN
          REDIM PUText$(1)
          PUText$(1) = "Test pit data has different radial offsets...  Skipping Test Pit analysis"
          CALL PopupError
          ExitCode% = 999
          EXIT SUB
        END IF
      NEXT I%
      ERASE Radial2
    CASE 4 TO 29, 32, 34 TO 36
      'do nothing
    CASE 30                             'active sequence drops
      Posit% = INSTR(LineData$, ".")
      ActiveDrops2% = Posit% - 1
      IF ActiveDrops2% <> ActiveDrops% THEN
        REDIM PUText$(1)
        PUText$(1) = "Test pit data has different number of drops...  Skipping Test Pit analysis"
        CALL PopupError
        ExitCode% = 999
        EXIT SUB
      END IF
    CASE 31                             'sequence heights
      Heights$ = LineData$
    CASE 33                             'peaks stored
      CheckText$ = LEFT$(LineData$, ActiveDrops%)
      InitNumPeaks2% = InCount2%(CheckText$, "*")
      IF InitNumPeaks2% <> InitNumPeaks% THEN
        REDIM PUText$(1)
        PUText$(1) = "Test pit data has different number of drops...  Skipping Test Pit analysis"
        CALL PopupError
        ExitCode% = 999
        EXIT SUB
      END IF
      DropNum2% = 0
      REDIM Heights2%(MaxNumPeaks%)
      FOR I% = 1 TO ActiveDrops%        'height for each drop in sequence
        StorePeak$ = MID$(CheckText$, I%, 1)
        IF StorePeak$ = "*" THEN
          HeightVal% = VAL(MID$(Heights$, I%, 1))
          IF HeightVal% <> 0 THEN
            DropNum2% = DropNum2% + 1
            Heights2%(DropNum2%) = HeightVal%
          END IF
        END IF
      NEXT I%
      FOR I% = 1 TO ActiveDrops%
        IF Heights%(I%) <> Heights2%(I%) THEN
          REDIM PUText$(1)
          PUText$(1) = "Test pit data has different drop heights...  Skipping Test Pit analysis"
          CALL PopupError
          ExitCode% = 999
          EXIT SUB
        END IF
      NEXT I%
      ERASE Heights2%
  END SELECT
END SUB

FUNCTION GammaLN (XX) STATIC
  REDIM COF#(6)
  COF#(1) = 76.18009173#: COF#(2) = -86.50532033#
  COF#(3) = 24.01409822#: COF#(4) = -1.231739516#
  COF#(5) = .00120858003#: COF#(6) = -.00000536382#
  X# = XX - 1#
  TMP# = X# + 5.5#
  TMP# = (X# + .5#) * LOG(TMP#) - TMP#
  SER# = 1#
  FOR J% = 1 TO 6
```

```
    X# = X# + 1#
    SER# = SER# + COF#(J%) / X#
  NEXT J%
  GammaLN = TMP# + LOG(2.506627465# * SER#)
END FUNCTION

SUB GetFileName (ExitCode%) STATIC
  STATIC ZP$
  WindowType% = 1: CLS
  IF ZP$ = "" THEN ZP$ = "N"
  IF PrnType$ = "" THEN PrnType$ = "L"
  WFile$ = File$
  IF Ext$ <> "" THEN
    WFile$ = WFile$ + Ext$
  END IF
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  FWD Data File Selection  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 1, 66: PRINT "Page 1"
  LOCATE 5, 17: PRINT "Deflection analyst:"
  LOCATE 8, 7: PRINT "Directory path for data file:";
  LOCATE 11, 7: PRINT "Do you want a list of data files for this path (Y/N) "
  LOCATE 14, 7: PRINT "Deflection Data File Name:"
  LOCATE 17, 7: PRINT "Printer type [(D)ot matrix/(L)aser]:"
  CALL HiliteColor
  LOCATE 5, 37: PRINT Analyst$
  LOCATE 8, 37: PRINT FPath$
  LOCATE 11, 60: PRINT ZP$
  LOCATE 14, 34: PRINT WFile$
  LOCATE 17, 44: PRINT PrnType$
  CALL NormalColor
  LOCATE 25, 4
  PRINT "         F10:Quit  "; CHR$(24); CHR$(25); "    Home  End        PgDn";
  Item% = 1
  MaxItem% = 5
  DO
    SELECT CASE Item%
      CASE 1
        CALL GetString(5, 37, 40, Analyst$, "L", 0, 0, "", Dummy%, ExitCode%)
      CASE 2
        OldPath$ = FPath$
        CALL GetString(8, 37, 40, FPath$, "L", 0, 0, "", Dummy%, ExitCode%)
        FPath$ = QPTrim$(UCASE$(FPath$))
        CurrDrive$ = CHR$(GetDrive%)
        CurrDir$ = GetDir$(CurrDrive$)
        CurrPath$ = CurrDrive$ + ":" + CurrDir$
        IF FPath$ <> "" THEN
          IF MID$(FPath$, 2, 1) = ":" THEN
            ChkDrive$ = LEFT$(FPath$, 1)
            IF NOT GoodDrive%(ChkDrive$) THEN       'check if valid drive
              REDIM PUText$(1)
              PUText$(1) = "Drive " + ChkDrive$ + " is not a valid choice... Please try another path."
              CALL PopupError
              ExitCode% = 0
              FPath$ = OldPath$
            ELSE                                    'drive OK, check dir
              IF RIGHT$(FPath$, 1) = "\" THEN
                FPath$ = LEFT$(FPath$, LEN(FPath$) - 1)
              END IF
              IF RIGHT$(FPath$, 1) <> ":" THEN
                FPath$ = FPath$ + "\"
              END IF
              CALL CDir(FPath$)
              IF NOT DOSError% THEN                 'path OK
                CALL CDir(CurrPath$)                '  switch back to curr dir
              ELSE                                  'path not OK
                REDIM PUText$(2)
                PUText$(1) = "Error occurred switching to " + FPath$
                PUText$(2) = "May not be a valid path... Please try again."
```

```
            CALL PopupError
            ExitCode% = 0
            FPath$ = OldPath$
          END IF
        END IF
      ELSE                              'no drive letter in specified path
        IF RIGHT$(FPath$, 1) = "\" THEN
          FPath$ = LEFT$(FPath$, LEN(FPath$) - 1)
        END IF
        CALL CDir(FPath$)
        IF NOT DOSError% THEN                   'path OK
          CALL CDir(CurrPath$)                  '  switch back to curr dir
        ELSE                                    'path not OK
          REDIM PUText$(2)
          PUText$(1) = "Error occurred switching to " + FPath$
          PUText$(2) = "May not be a valid path... Please try again."
          CALL PopupError
          ExitCode% = 0
          FPath$ = OldPath$
        END IF
      END IF
    END IF
    IF FPath$ <> "" AND RIGHT$(FPath$, 1) <> "\" THEN FPath$ = FPath$ + "\"
    LOCATE 8, 37: PRINT FPath$
  CASE 3
    OldZP$ = ZP$
    CALL GetString(11, 60, 1, ZP$, "L", 0, 0, "", Dummy%, ExitCode%)
    IF ExitCode% <> 68 THEN     'quit
      ZP$ = UCASE$(ZP$)
      SELECT CASE ZP$
        CASE "Y"
          ShowFiles$ = FPath$ + "*.PKS"
          NumMatchs% = FCount%(ShowFiles$)
          IF NumMatchs% > 0 THEN
            CALL DisplayFileNames(NumMatchs%, ShowFiles$, FPath$, File$, Ext$, ExitCode%, 0)
            WFile$ = File$ + Ext$
          ELSE
            REDIM PUText$(1)
            PUText$(1) = "No files found matching " + ShowFiles$
            CALL PopupError
            ZP$ = "N"
            LOCATE 11, 60: PRINT ZP$
            ExitCode% = 72          'go back to PATH field!
          END IF
        CASE "N"
          'go on
        CASE ELSE
          REDIM PUText$(1)
          PUText$(1) = "Please enter a Y or N ONLY..."
          CALL PopupError
          ExitCode% = 0
      END SELECT
    END IF
  CASE 4
    DO
      OldWFile$ = WFile$
      CALL GetString(14, 34, 12, WFile$, "L", 0, 0, "", Dummy%, ExitCode%)
      IF ExitCode% = 68 THEN EXIT DO    'quit
      WFile$ = QPTrim$(UCASE$(WFile$))
      LF% = LEN(WFile$)
      FOR VV% = 1 TO LF%
        IF ASC(MID$(WFile$, VV%, 1)) = 32 THEN
          REDIM PUText$(1)
          PUText$(1) = "Spaces are NOT allowed in file names..."
          CALL PopupError
          WFile$ = OldWFile$
          ExitCode% = 0
          EXIT FOR
        END IF
      NEXT VV%
      IF ExitCode% <> 0 THEN
        SP% = INSTR(WFile$, ".")
        IF SP% <> 0 THEN
```

```
            File$ = LEFT$(WFile$, SP% - 1)
            Ext$ = QPTrim$(RIGHT$(WFile$, LEN(WFile$) - (SP% - 1)))
          ELSE
            File$ = QPTrim$(LEFT$(WFile$, 8))
            Ext$ = ""
          END IF
          IF Ext$ <> ".PKS" THEN
            REDIM PUText$(1)
            PUText$(1) = "Please specify a file with an extension of '.PKS' ..."
            CALL PopupError
            WFile$ = OldWFile$
            ExitCode% = 0
          ELSE
            EXIT DO
          END IF
        END IF
      LOOP
    CASE 5
      OldType$ = PrnType$
      DO
        CALL GetString(17, 44, 1, PrnType$, "L", 0, 0, "", Dummy%, ExitCode%)
        IF ExitCode% <> 68 THEN     'quit
          PrnType$ = UCASE$(PrnType$)
          SELECT CASE PrnType$
            CASE "D", "L"
              EXIT DO
            CASE ELSE
              REDIM PUText$(1)
              PUText$(1) = "Please enter a D or L ONLY..."
              CALL PopupError
              ExitCode% = 0
              PrnType$ = OldType$
          END SELECT
        END IF
      LOOP
  END SELECT
  SELECT CASE ExitCode%                          'determine next action
    CASE 71                   'home
      Item% = 1
    CASE 79                   'end
      Item% = MaxItem%
    CASE 15, 72               'Shift-Tab, up arrow
      Item% = Item% - 1
    CASE 9, 13, 80            'Tab, CR, down arrow
      Item% = Item% + 1
    CASE 81                   'PgDn
      IF File$ = "" THEN
        REDIM PUText$(1)
        PUText$(1) = "A file name MUST be entered... please try again!"
        CALL PopupError
        Item% = 4
      ELSE
        ChkName$ = FPath$ + File$ + Ext$
        IF NOT Exist%(ChkName$) THEN
          REDIM PUText$(1)
          PUText$(1) = "File not found...  Please try again."
          CALL PopupError
          File$ = ""
          Ext$ = ""
          ExitCode% = 0
          Item% = 3
        ELSE
          ExitCode% = 1
          SELECT CASE PrnType$
            CASE "D"
              ScrnDumpRes$ = ""
            CASE "L"
              ScrnDumpRes$ = "100"
          END SELECT
          EXIT SUB
        END IF
      END IF
    CASE 68                   'F10: quit
```

```
        CALL Quit
      CASE ELSE
        'do nothing
    END SELECT
    IF Item% < 1 THEN Item% = 1
    IF Item% > MaxItem% THEN Item% = MaxItem%
  LOOP
END SUB

SUB GetMaterialProps STATIC
  WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  Material Properties Control Information  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 5, 10: PRINT "Number of pavement layers above the subgrade:"
  LOCATE 7, 43: PRINT "  Layer              SHRP"
  LOCATE 8, 43: PRINT "Thickness           Material"
  LOCATE 9, 43: PRINT "(inches)              Code"
  LOCATE 10, 43: PRINT "---------           --------"
  LOCATE 25, 4
  PRINT "                    "; CHR$(24); CHR$(25); CHR$(27); CHR$(26); "  Home  End        PgDn";
  FOR I% = 1 TO MaxLayers%
    LOCATE 10 + I%, 12
    IF I% <= NumLayers% THEN
      CALL NormalColor
      PRINT "Layer "; QPRTrim$(STR$(I%));
      IF I% = 1 THEN
        PRINT "   (surface)";
      ELSEIF I% = NumLayers% AND NumLayers% > 1 THEN
        PRINT "   (on subgrade)";
      ELSE
        PRINT SPACE$(16);
      END IF
      CALL HiliteColor
      LOCATE , 45: PRINT DefaultReal$(Thickness(I%), "R", 5, "###.#");
      LOCATE , 65: PRINT DefaultInt$(MatlCode%(I%), "R", 3)
    ELSE
      PRINT SPACE$(65)
    END IF
  NEXT I%
  CALL HiliteColor
  LOCATE 5, 56: PRINT DefaultInt$(NumLayers%, "R", 2)
  Item% = 1
  MaxItem% = 2 * NumLayers% + 1
  DO
    SELECT CASE Item%
      CASE 1
        OldLayers% = NumLayers%
        CALL GetInt(5, 56, 2, NumLayers%, "R", 0, 0, "", Dummy%, ExitCode%)
        IF NumLayers% < 1 OR NumLayers% > 10 THEN
          REDIM PUText$(1)
          PUText$(1) = "Please enter a value between 1 and 10 ONLY..."
          CALL PopupError
          NumLayers% = OldLayers%
          ExitCode% = 0
        ELSEIF NumLayers% >= 1 AND NumLayers% <> OldLayers% THEN
          MaxItem% = 2 * NumLayers% + 1
          FOR I% = 1 TO MaxLayers%
            LOCATE 10 + I%, 12
            IF I% <= NumLayers% THEN
              CALL NormalColor
              PRINT "Layer "; QPRTrim$(STR$(I%));
              IF I% = 1 THEN
                PRINT "   (surface)";
              ELSEIF I% = NumLayers% AND NumLayers% > 1 THEN
                PRINT "   (on subgrade)";
              ELSE
                PRINT SPACE$(16);
              END IF
```

```
            CALL HiliteColor
            LOCATE , 45: PRINT DefaultReal$(Thickness(I%), "R", 5, "###.#");
            LOCATE , 65: PRINT DefaultInt$(MatlCode%(I%), "R", 3)
          ELSE
            PRINT SPACE$(65)
          END IF
        NEXT I%
      END IF
    CASE 2, 4, 6, 8, 10, 12, 14, 16, 18, 20
      CALL NormalColor
      LOCATE 25, 4: PRINT "                    ";
      Element% = Item% \ 2
      OldThickness = Thickness(Element%)
      CALL GetReal(10 + Element%, 45, 5, "###.#", Thickness(Element%), "L", 0, 0, "", Dummy%, ExitCode%)
      IF Thickness(Element%) <= 0 THEN
        REDIM PUText$(1)
        PUText$(1) = "Please enter a number greater than 0 ONLY..."
        CALL PopupError
        ExitCode% = 0
        Thickness(Element%) = OldThickness
      ELSEIF Thickness(Element%) > 24 THEN
        REDIM PUText$(2)
        PUText$(1) = "Thicknesses greater than 24 inches are very unusual."
        PUText$(2) = "   Please verify that this thickness is correct..."
        CALL PopupWarning(0, 0, "")
      END IF
    CASE 3, 5, 7, 9, 11, 13, 15, 17, 19, 21
      CALL NormalColor
      LOCATE 25, 4: PRINT "F3:MaterialsList   ";
      Element% = (Item% - 1) \ 2
      OldCode% = MatlCode%(Element%)
      CALL GetInt(10 + Element%, 65, 3, MatlCode%(Element%), "R", 0, 0, "", Dummy%, ExitCode%)
      IF LEN(QPTrim$(STR$(MatlCode%(Element%)))) <> 3 AND ExitCode% <> 61 THEN
        REDIM PUText$(1)
        PUText$(1) = "Please enter a 3-digit material code ONLY..."
        CALL PopupError
        ExitCode% = 0
      ELSEIF ExitCode% = 61 THEN          'F3: Database list
        IF OldCode% = 0 THEN
          SelectedCode% = 1
        ELSE
          TmpCode$ = QPTrim$(STR$(OldCode%))
          GOSUB GetCode
        END IF
        CALL PickFromLongList(" List of Materials ", MaterialList$(), SelectedCode%, 3, 2, 24, 35, ExitCode%, 0)
      ELSE
        TmpCode$ = QPTrim$(STR$(MatlCode%(Element%)))
        GOSUB GetCode
        IF NumMaterials% = -1 THEN
          REDIM PUText$(1)
          PUText$(1) = "Invalid code.  Please try again..."
          CALL PopupError
          ExitCode% = 0
          MatlCode%(Element%) = OldCode%
        END IF
      END IF
      IF Element% = 1 AND ExitCode% <> 0 AND NOT (SelectedCode% = 20 OR SelectedCode% = 21) THEN
        REDIM PUText$(1)
        PUText$(1) = "Invalid code for layer 1.  Please try again..."
        CALL PopupError
        ExitCode% = 0
        MatlCode%(Element%) = OldCode%
      END IF
      IF ExitCode% <> 0 THEN
        MatlCode%(Element%) = VAL(MaterialList$(SelectedCode%))
        Modulus(Element%) = ModulusList(SelectedCode%)
        PoissonRatio(Element%) = PoissonList(SelectedCode%)
        ALo(Element%) = AListLo(SelectedCode%)
        AHi(Element%) = AListHi(SelectedCode%)
        SelCode%(Element%) = SelectedCode%
      END IF
      CALL HiliteColor
      LOCATE 10 + Element%, 65: PRINT DefaultInt$(MatlCode%(Element%), "R", 3)
```

```
    END SELECT
    SELECT CASE ExitCode%                          'determine next action
      CASE 71                  'home
        Item% = 1
      CASE 79                  'end
        Item% = MaxItem%
      CASE 15, 75              'Shift-Tab, left arrow
        Item% = Item% - 1
      CASE 72                  'up arrow
        Item% = Item% - 2
      CASE 9, 13, 77           'Tab, CR, right arrow
        Item% = Item% + 1
      CASE 80                  'down arrow
        IF Item% = 1 THEN
          Item% = Item% + 1
        ELSE
          Item% = Item% + 2
        END IF
      CASE 81            'PgDn
        SetMaterials% = True%
        FOR aaa% = 1 TO NumLayers%
          IF Modulus(aaa%) = 0 THEN SetMaterials% = False%: EXIT FOR
        NEXT aaa%
        IF SetMaterials% THEN
          IF MatlCode%(1) = 700 THEN
            CALL GetTemperatureGradientsControl
            CALL CalcMidDepthTemps
            CALL CalcTempCorrDefl
            CALL CalcCorrStationStats
            Corrected% = True%
          ELSE                 'if no correction applied, assign norm defls
            REDIM MeanCorrDefl(NumStations%, MaxHeight%)
            FOR I% = 1 TO NumStations%
              FOR J% = 1 TO MaxHeight%
                MeanCorrDefl(I%, J%) = MeanNormDefl(I%, J%, 1)
              NEXT J%
            NEXT I%
            IF NumTPStations% > 0 THEN
              REDIM TPMeanCorrDefl(NumTPStations%, MaxHeight%)
              FOR I% = 1 TO NumTPStations%
                FOR J% = 1 TO MaxHeight%
                  TPMeanCorrDefl(I%, J%) = TPMeanNormDefl(I%, J%, 1)
                NEXT J%
              NEXT I%
            END IF
          END IF
          ExitCode% = 1
          EXIT SUB
        ELSE
          REDIM PUText$(1)
          PUText$(1) = "Material codes MUST be entered for EVERY layer..."
          CALL PopupError
        END IF
      CASE ELSE
        'do nothing
    END SELECT
    IF Item% < 1 THEN Item% = 1
    IF Item% > MaxItem% THEN Item% = MaxItem%
  LOOP

GetCode:
  NumMaterials% = 20
  CALL Find(BYVAL VARPTR(MaterialList$(1)), NumMaterials%, TmpCode$)
  IF NumMaterials% = -1 THEN
    SelectedCode% = 1
  ELSE
    SelectedCode% = NumMaterials% + 1
  END IF
  RETURN

END SUB

SUB GetRunningComments STATIC
```

```
  SCREEN 0: WIDTH 80, 25: WindowType% = 1: CLS
  CALL GetText(Comment$, 1, 1, 25, 80, 2, 0, ExitCode%)
END SUB

SUB GetSubsectionBoundaries (ExitCode%) STATIC
  WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  Subsection Boundary Definition  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 5, 7: PRINT "Number of subsections: ";
  LOCATE 4, 57: PRINT "Number of"
  LOCATE 5, 59: PRINT "Tests     Status"
  LOCATE 25, 23
  PRINT CHR$(24); CHR$(25); "    Home  End        PgDn";
  SubSectEnd%(NumSubSect%) = 500
  IF NumSubSect% >= 1 THEN
    FOR I% = 1 TO NumSubSect%
      CALL NormalColor
      LOCATE 5 + 2 * I%, 12: PRINT "Station for end of Subsection"; QPRTrim$(STR$(I%)); ":     ft.";
      IF I% <> NumSubSect% THEN CALL HiliteColor
      LOCATE , 45: PRINT QPTrim$(STR$(SubSectEnd%(I%)))
    NEXT I%
    CALL CalcStationsInSubsection
  END IF
  CALL HiliteColor
  LOCATE 5, 30: PRINT QPTrim$(STR$(NumSubSect%))
  Item% = 1
  MaxItem% = NumSubSect%
  DO
    SELECT CASE Item%
      CASE 1
        OldNumSubSect% = NumSubSect%
        CALL GetInt(5, 30, 1, NumSubSect%, "L", 0, 0, "", Dummy%, ExitCode%)
        IF NumSubSect% < 1 OR NumSubSect% > 5 THEN
          REDIM PUText$(1)
          PUText$(1) = "Please enter a number from 1 to 5 only..."
          CALL PopupError
          ExitCode% = 0
          NumSubSect% = OldNumSubSect%
        ELSEIF NumSubSect% >= 1 THEN
          MaxItem% = NumSubSect%
          SubSectEnd%(NumSubSect%) = 500
          FOR I% = 1 TO 5
            IF I% <= NumSubSect% THEN
              CALL NormalColor
              LOCATE 5 + 2 * I%, 12: PRINT "Station for end of Subsection"; QPRTrim$(STR$(I%)); ":     ft.";
              IF I% <> NumSubSect% THEN CALL HiliteColor
              LOCATE , 45: PRINT QPTrim$(STR$(SubSectEnd%(I%)))
            ELSE
              LOCATE 5 + 2 * I%, 12: PRINT SPACE$(66)
            END IF
          NEXT I%
        END IF
      CASE 2 TO NumSubSect% + 1
        Element% = Item% - 1
        OldEnd% = SubSectEnd%(Element%)
        CALL GetInt(5 + 2 * Element%, 45, 3, SubSectEnd%(Element%), "L", 0, 0, "", Dummy%, ExitCode%)
        IF Element% = 1 THEN
          IF SubSectEnd%(Element%) <= 0 THEN
            REDIM PUText$(1)
            PUText$(1) = "Please enter a number greater than 0 ONLY..."
            CALL PopupError
            ExitCode% = 0
            SubSectEnd%(Element%) = OldEnd%
          END IF
        ELSE
          IF SubSectEnd%(Element%) < SubSectEnd%(Element% - 1) OR SubSectEnd%(Element%) > 500 THEN
            REDIM PUText$(1)
            PUText$(1) = "Please enter a number between " + STR$(SubSectEnd%(Element% - 1)) + " and 500 ONLY..."
```

```
            CALL PopupError
            ExitCode% = 0
            SubSectEnd%(Element%) = OldEnd%
          END IF
        END IF
    END SELECT
    CALL CalcStationsInSubsection
    SELECT CASE ExitCode%                          'determine next action
      CASE 71           'home
        Item% = 1
      CASE 79           'end
        Item% = MaxItem%
      CASE 15, 72         'Shift-Tab, up arrow
        Item% = Item% - 1
      CASE 9, 13, 80      'Tab, CR, down arrow
        Item% = Item% + 1
      CASE 81           'PgDn
        FOR I% = 1 TO NumSubSect%
          IF StationCount%(I%) < MinPointsInSubSection% THEN
            ExitCode% = 0
            Item% = I% + 1
            EXIT FOR
          END IF
        NEXT I%
        IF ExitCode% <> 0 THEN
          ExitCode% = 1
          EXIT SUB
        END IF
      CASE ELSE
        'do nothing
    END SELECT
    IF Item% < 1 THEN Item% = 1
    IF Item% > MaxItem% THEN Item% = MaxItem%
  LOOP
END SUB

SUB GetTemperatureGradientsControl STATIC
  TPage% = 1
  DO
    SELECT CASE TPage%
      CASE 1
        CALL GetTemperatureGradientsPage1(ExitCode%)
      CASE 2
        CALL GetTemperatureGradientsPage2(1, ExitCode%)
      CASE 3
        CALL GetTemperatureGradientsPage2(2, ExitCode%)
    END SELECT
    SELECT CASE ExitCode%
      CASE 1
        TPage% = TPage% + 1
      CASE 0
      CASE -1
        TPage% = TPage% - 1
    END SELECT
  LOOP WHILE TPage% <= 3
END SUB

SUB GetTemperatureGradientsPage1 (ExitCode%) STATIC
  WIDTH 80, 25: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  Asphalt Surfaced Pavement Temperature Gradient Control Data  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 4, 10: PRINT "   Number of depths for temperature at each time:"
  LOCATE 5, 10: PRINT "Number of times temperature measured first area:"
  LOCATE 6, 10: PRINT "Number of times temperature measured second area:"
  LOCATE 8, 27: PRINT "Test Area                Test Area"
  LOCATE 9, 27: PRINT "sta < 0+00               sta > 5+00"
  LOCATE 10, 27: PRINT "----------               ----------"
  LOCATE 25, 4
```

```
PRINT "                    "; CHR$(24); CHR$(25); "    Home  End        PgDn";
GOSUB RedrawTables1
CALL HiliteColor
LOCATE 4, 62: PRINT DefaultInt$(NumDepths%, "R", 2)
LOCATE 5, 62: PRINT DefaultInt$(NumTimes%(1), "R", 2)
LOCATE 6, 62: PRINT DefaultInt$(NumTimes%(2), "R", 2)
Item% = 1
MaxItem% = NumTimes%(1) + NumTimes%(2) + 3
DO
  SELECT CASE Item%
    CASE 1
      OldDepths% = NumDepths%
      CALL GetInt(4, 62, 2, NumDepths%, "R", 0, 0, "", Dummy%, ExitCode%)
      IF NumDepths% < 2 OR NumDepths% > 3 THEN
        REDIM PUText$(1)
        PUText$(1) = "Please enter a 2 or 3 ONLY..."
        CALL PopupError
        NumDepths% = OldDepths%
        ExitCode% = 0
      ELSEIF NumDepths% <> OldDepths% THEN
        GOSUB RedrawTables1
      END IF
    CASE 2
      OldTime% = NumTimes%(1)
      CALL GetInt(5, 62, 2, NumTimes%(1), "R", 0, 0, "", Dummy%, ExitCode%)
      IF NumTimes%(1) < 2 OR NumTimes%(1) > 10 THEN
        REDIM PUText$(1)
        PUText$(1) = "Please enter a value between 2 and 10 ONLY..."
        CALL PopupError
        NumTimes%(1) = OldTime%
        ExitCode% = 0
      ELSEIF NumTimes%(1) <> OldTime% THEN
        GOSUB RedrawTables1
      END IF
    CASE 3
      OldTime% = NumTimes%(2)
      CALL GetInt(6, 62, 2, NumTimes%(2), "R", 0, 0, "", Dummy%, ExitCode%)
      IF NumTimes%(2) < 2 OR NumTimes%(2) > 10 THEN
        REDIM PUText$(1)
        PUText$(1) = "Please enter a value between 2 and 10 ONLY..."
        CALL PopupError
        NumTimes%(2) = OldTime%
        ExitCode% = 0
      ELSEIF NumTimes%(2) <> OldTime% THEN
        GOSUB RedrawTables1
      END IF
    CASE 4 TO (NumTimes%(1) + 3)
      J% = 1
      Element% = Item% - 3
      OldTime$ = MeasTime$(Element%, J%)
      CALL GetString(10 + Element%, 30, 4, MeasTime$(Element%, J%), "L", 0, 0, "", Dummy%, ExitCode%)
      MeasTime$(Element%, J%) = QPTrim$(MeasTime$(Element%, J%))
      GOSUB ValidateTime
    CASE (NumTimes%(1) + 4) TO (NumTimes%(1) + NumTimes%(2) + 3)
      J% = 2
      Element% = Item% - (NumTimes%(1) + 3)
      OldTime$ = MeasTime$(Element%, J%)
      CALL GetString(10 + Element%, 55, 4, MeasTime$(Element%, J%), "L", 0, 0, "", Dummy%, ExitCode%)
      MeasTime$(Element%, J%) = QPTrim$(MeasTime$(Element%, J%))
      GOSUB ValidateTime
  END SELECT
  SELECT CASE ExitCode%                          'determine next action
    CASE 71                  'home
      Item% = 1
    CASE 79                  'end
      Item% = MaxItem%
    CASE 72                  'up arrow
      Item% = Item% - 1
    CASE 80, 13              'down arrow,CR
      Item% = Item% + 1
    CASE 81             'PgDn
      OKToExit% = True%
      IF NumTimes%(1) < 2 OR NumTimes%(2) < 2 THEN OKToExit% = False%
```

```
        FOR J% = 1 TO 2
          FOR I% = 1 TO NumTimes%(J%)
            IF MeasTime$(I%, J%) <> "" THEN
              MeasMinutes%(I%, J%) = VAL(RIGHT$(MeasTime$(I%, J%), 2)) + 60 * VAL(LEFT$(MeasTime$(I%, J%), 2))
            ELSE
              REDIM PUText$(1)
              PUText$(1) = "Please enter a value for all times before moving to the next page..."
              CALL PopupError
              OKToExit% = False%
              EXIT FOR
            END IF
          NEXT I%
          IF NOT OKToExit% THEN EXIT FOR
        NEXT J%
        IF OKToExit% THEN
          CrossingMidnight% = False%
          FOR J% = 1 TO 2
            FOR I% = 2 TO NumTimes%(J%)
              IF MeasMinutes%(I%, J%) <= MeasMinutes%(I% - 1, J%) THEN
                IF MeasMinutes%(I% - 1, J%) >= 1320 AND MeasMinutes%(I%, J%) <= 120 THEN
                  CrossingMidnight% = True%
                  CrossingInterval%(J%) = I% - 1
                ELSE
                  'only an error if EITHER time is outside 10PM to 2AM window
                  REDIM PUText$(5)
                  PUText$(1) = "Values for all times MUST be entered in sequence except  "
                  PUText$(2) = "when crossing the midnight boundary (special case ONLY). "
                  PUText$(3) = ""
                  PUText$(4) = "In order to be considered 'crossing midnight', both times"
                  PUText$(5) = "MUST be in the period between 10 PM and 2 AM ONLY..."
                  CALL PopupError
                  OKToExit% = False%
                  EXIT FOR
                END IF
              END IF
            NEXT I%
            IF NOT OKToExit% THEN EXIT FOR
          NEXT J%
          IF OKToExit% THEN
            ExitCode% = 1
            EXIT SUB
          END IF
        END IF
      CASE ELSE
        'do nothing
    END SELECT
    IF Item% < 1 THEN Item% = 1
    IF Item% > MaxItem% THEN Item% = MaxItem%
  LOOP
  EXIT SUB

RedrawTables1:
  MaxItem% = NumTimes%(1) + NumTimes%(2) + 3
  FOR J% = 1 TO 2
    FOR I% = 1 TO MaxTimes%
      LOCATE I% + 10, 25 * J% - 5
      IF I% <= NumTimes%(J%) THEN
        CALL NormalColor
        PRINT " Time "; QPTrim$(STR$(I%)); ":";
        CALL HiliteColor
        LOCATE , 5 + 25 * J%: PRINT MeasTime$(I%, J%);
      ELSE
        PRINT SPACE$(25)
      END IF
    NEXT I%
  NEXT J%
  RETURN

ValidateTime:
  IF LEN(MeasTime$(Element%, J%)) <> 4 THEN
    REDIM PUText$(1)
    PUText$(1) = "Please enter a time in HHMM format ONLY..."
    CALL PopupError
```

```
    ExitCode% = 0
    MeasTime$(Element%, J%) = OldTime$
  ELSE
    Hour% = VAL(LEFT$(MeasTime$(Element%, J%), 2))
    Minute% = VAL(RIGHT$(MeasTime$(Element%, J%), 2))
    IF Hour% < 0 OR Hour% > 23 OR Minute% < 0 OR Minute% > 59 THEN
      REDIM PUText$(1)
      PUText$(1) = "Please enter a time between 0000 and 2359 ONLY..."
      CALL PopupError
      ExitCode% = 0
      MeasTime$(Element%, J%) = OldTime$
    END IF
  END IF
  RETURN

END SUB

SUB GetTemperatureGradientsPage2 (J%, ExitCode%) STATIC
  WIDTH 80, 25: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  Asphalt Surfaced Pavement Temperature Gradient Data  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 5, 35: PRINT "Temperature (°F) Measurement Area"; J%
  LOCATE 6, 46
  IF J% = 1 THEN
    PRINT "station < 0+00"
  ELSE
    PRINT "station > 5+00"
  END IF
  LOCATE 8, 39: PRINT "Depth 1   Depth 2   Depth 3"
  LOCATE 9, 39: PRINT "-------   -------   -------"
  LOCATE 25, 4
  IF J% = 1 THEN
    PRINT "                    "; CHR$(24); CHR$(25); "    Home  End  PgUp  PgDn";
  ELSE
    PRINT "                    "; CHR$(24); CHR$(25); "    Home  End        PgDn";
  END IF
  FOR I% = 1 TO NumTimes%(J%)
    LOCATE I% + 10, 21
    CALL NormalColor
    PRINT " Time "; QPTrim$(STR$(I%)); ":";
    LOCATE , 30: PRINT MeasTime$(I%, J%);
    CALL HiliteColor
    FOR K% = 1 TO NumDepths%
      LOCATE , 30 + 10 * K%: PRINT DefaultReal$(TimeTemp(I%, J%, K%), "R", 5, "###.#");
    NEXT K%
    PRINT
  NEXT I%
  MaxRow% = NumTimes%(J%)
  MaxCol% = NumDepths%
  CCol% = 1
  CRow% = 1
  DO
    SRow% = 10 + CRow%
    SCol% = 30 + CCol% * 10
    OldTimeTemp = TimeTemp(CRow%, J%, CCol%)
    CALL GetReal(SRow%, SCol%, 5, "###.#", TimeTemp(CRow%, J%, CCol%), "L", 0, 0, "", Dummy%, ExitCode%)
    IF TimeTemp(CRow%, J%, CCol%) < 0 OR TimeTemp(CRow%, J%, CCol%) > 160 THEN
      REDIM PUText$(1)
      PUText$(1) = "Please enter a temperature between 0°F and 160°F ONLY..."
      CALL PopupError
      TimeTemp(CRow%, J%, CCol%) = OldTimeTemp
      ExitCode% = 0
    END IF
    SELECT CASE ExitCode%                          'determine next action
      CASE 71           'home
        CRow% = 1
        CCol% = 1
      CASE 79           'end
```

```
        CRow% = MaxRow%
        CCol% = MaxCol%
      CASE 15, 75       'Shift-Tab, left arrow
        CCol% = CCol% - 1
      CASE 72           'up arrow
        CRow% = CRow% - 1
      CASE 9, 77        'Tab, right arrow
        CCol% = CCol% + 1
      CASE 80           'down arrow
        CRow% = CRow% + 1
      CASE 13           'CR
        CRow% = CRow% + 1
        IF CRow% > MaxRow% THEN
          CRow% = 1
          CCol% = CCol% + 1
          IF CCol% > MaxCol% THEN CCol% = 1
        END IF
      CASE 81           'PgDn
        ExitCode% = 1
        EXIT DO
      CASE 73           'PgUp
        ExitCode% = -1
        EXIT DO
      CASE ELSE
        'do nothing
    END SELECT
    IF CCol% < 1 THEN CCol% = MaxCol%
    IF CCol% > MaxCol% THEN CCol% = 1
    IF CRow% < 1 THEN CRow% = MaxRow%
    IF CRow% > MaxRow% THEN CRow% = 1
  LOOP
END SUB

SUB OutliersMenu (ExitCode%) STATIC
  SCREEN 0: WIDTH 80, 25: CLS
  Choice% = 1
  CALL CalcOutliers
  DO
    REDIM Item$(5)
    Title$ = "OUTLIERS ANALYSIS MENU"
    Item$(1) = "View Outlier Analysis Curves"
    Item$(2) = "Add to Running Comments"
    Item$(3) = "Go on to STRUCTURAL Analysis"
    Item$(4) = "Select NEW Data File"
    Item$(5) = "QUIT Program"
    CALL BarMenu(Title$, Item$(), Choice%, 0)
    SELECT CASE Choice%
      CASE 1
        CALL GetOutlierCurveParameters(ExitCode%)
        Choice% = 2
      CASE 2
        CALL GetRunningComments
        Choice% = 3
      CASE 3
        ExitCode% = 1
        EXIT DO
      CASE 4
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteRunningComments
        CLOSE #2
        ExitCode% = -2
        EXIT DO
      CASE 5
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteRunningComments
        CLOSE #2
        CALL Quit
    END SELECT
  LOOP
END SUB
```

```
SUB ReadFWDDataFile STATIC
  DO
    IF LineCounter& < 37 THEN
      CALL CheckHeader
    ELSE
      CALL ReadNextLine(DataType%)
      SELECT CASE DataType%
        CASE 1                          'peak deflection data block
          CALL ReadPeaks
        CASE -1
          EXIT DO
        CASE ELSE
          'do nothing
      END SELECT
    END IF
  LOOP
END SUB

SUB ReadNextLine (DataType%) STATIC
  IF NOT EOF(1) THEN
    LINE INPUT #1, LineData$
    DataType$ = LEFT$(LineData$, 1)
    DataType% = INSTR("SB'E*- 1234567890", DataType$)
    LineCounter& = LineCounter& + 1
    IF DataType% = 4 THEN
      IF UCASE$(LEFT$(LineData$, 3)) = "EOF" THEN
        DataType% = -1
      END IF
    END IF
  ELSE
    DataType% = -1                      'end of file occurred
  END IF
END SUB

SUB ReadPeaks
' Process start of sensor data block:
  TestStation$ = QPTrim$(MID$(LineData$, 2, 8))
  LaneSpec$ = UCASE$(QPTrim$(MID$(LineData$, 10, 4)))
  PvmtType$ = LEFT$(LaneSpec$, 1)
  TPL% = VAL(RIGHT$(LaneSpec$, 1))
  TestTime$ = MID$(LineData$, 29, 4)
  ETemp$ = QPTrim$(MID$(LineData$, 37, 4))
  TestTemp = VAL(ETemp$)
  IF TPL% = 0 THEN
    NumTPStations% = NumTPStations% + 1
    TPStation%(NumTPStations%) = VAL(TestStation$)
    TPFWDMinutes%(NumTPStations%) = VAL(RIGHT$(TestTime$, 2)) + 60 * VAL(LEFT$(TestTime$, 2))
  ELSE
    NumStations% = NumStations% + 1
    PointLocation%(NumStations%) = TPL%
    Station%(NumStations%) = VAL(TestStation$)
    FWDMinutes%(NumStations%) = VAL(RIGHT$(TestTime$, 2)) + 60 * VAL(LEFT$(TestTime$, 2))
  END IF
  LocationList%(TPL% + 1) = True%       'list of valid test locations
  FOR I% = 1 TO InitNumPeaks%
    CALL ReadNextLine(DataType%)
    SELECT CASE DataType%
      CASE 1                            'found start of next peaks block
        CALL ReadPeaks
        EXIT SUB
      CASE ELSE                         'normal processing
        IF NOT English% THEN
          IF TPL% = 0 THEN               'test pit data
            TPTestLoad(NumTPStations%, I%) = VAL(MID$(LineData$, 1, 4))
            FOR J% = 1 TO NumDeflectors%
              Posit% = J% * 4 + 1
              TPDefl(NumTPStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 4))
            NEXT J%
          ELSE                          'NOT test pit data
            TestLoad(NumStations%, I%) = VAL(MID$(LineData$, 1, 4))
            FOR J% = 1 TO NumDeflectors%
              Posit% = J% * 4 + 1
              Defl(NumStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 4))
```

```
          NEXT J%
        END IF
      ELSE
        IF TPL% = 0 THEN
          TPTestLoad(NumTPStations%, I%) = VAL(MID$(LineData$, 33, 6))
          FOR J% = 1 TO NumDeflectors%
            Posit% = J% * 6 + 33
            TPDefl(NumTPStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 6))
          NEXT J%
        ELSE
          TestLoad(NumStations%, I%) = VAL(MID$(LineData$, 33, 6))
          FOR J% = 1 TO NumDeflectors%
            Posit% = J% * 6 + 33
            Defl(NumStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 6))
          NEXT J%
        END IF
      END IF
    END SELECT
  NEXT I%
END SUB

SUB ReadPeaks2
' Process start of sensor data block: for TEST PIT DATA FILES ONLY
  TestStation$ = QPTrim$(MID$(LineData$, 2, 8))
  LaneSpec$ = UCASE$(QPTrim$(MID$(LineData$, 10, 4)))
  PvmtType$ = LEFT$(LaneSpec$, 1)
  TPL% = VAL(RIGHT$(LaneSpec$, 1))
  TestTime$ = MID$(LineData$, 29, 4)
  ETemp$ = QPTrim$(MID$(LineData$, 37, 4))
  TestTemp = VAL(ETemp$)
  IF TPL% = 0 THEN
    NumTPStations% = NumTPStations% + 1
    TPStation%(NumTPStations%) = VAL(TestStation$)
    TPFWDMinutes%(NumTPStations%) = VAL(RIGHT$(TestTime$, 2)) + 60 * VAL(LEFT$(TestTime$, 2))
  ELSE
    EXIT SUB
  END IF
  LocationList%(1) = True%              'list of valid test locations
  FOR I% = 1 TO InitNumPeaks%
    CALL ReadNextLine(DataType%)
    SELECT CASE DataType%
      CASE 1                            'found start of next peaks block
        CALL ReadPeaks2
        EXIT SUB
      CASE ELSE                         'normal processing
        IF NOT English% THEN
          TPTestLoad(NumTPStations%, I%) = VAL(MID$(LineData$, 1, 4))
          FOR J% = 1 TO NumDeflectors%
            Posit% = J% * 4 + 1
            TPDefl(NumTPStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 4))
          NEXT J%
        ELSE
          TPTestLoad(NumTPStations%, I%) = VAL(MID$(LineData$, 33, 6))
          FOR J% = 1 TO NumDeflectors%
            Posit% = J% * 6 + 33
            TPDefl(NumTPStations%, I%, J%) = VAL(MID$(LineData$, Posit%, 6))
          NEXT J%
        END IF
    END SELECT
  NEXT I%
END SUB

SUB ReviewMenu STATIC
  SCREEN 0: WIDTH 80, 25: CLS
  Choice% = 1
  DO
    IF SummaryOfResults2% THEN
      REDIM Item$(6)
      Title$ = "REVIEW PREVIOUS DATA MENU"
      Item$(1) = "View Deflection vs Station Plot(s)"
      Item$(2) = "View Corr. Deflection vs Station Plot(s)"
      Item$(3) = "View Outlier Analysis Curves"
      Item$(4) = "Re-define Subsection Limits"
```

```
      Item$(5) = "Add to Running Comments"
      Item$(6) = "Return to Structural Analysis Menu"
      CALL BarMenu(Title$, Item$(), Choice%, 0)
      SELECT CASE Choice%
        CASE 1
          CALL GetUniformityCurveParameters
          Choice% = 2
        CASE 2
          CALL GetCorrectedDeflectionParameters
          Choice% = 3
        CASE 3
          CALL GetOutlierCurveParameters(ExitCode%)
          Choice% = 4
        CASE 4
          DO
            CALL GetSubsectionBoundaries(ExitCode%)
            CALL CalcSubSectionAlpha(Recalc%)
          LOOP WHILE Recalc%
          CALL CalcSubSectionStats
          CALL CalcOutliers
          SELECT CASE MatlCode%(1)
            CASE 700 'ACC
              CALL CalcFlexStats
            CASE 730 'PCC
              CALL CalcRigidStats
          END SELECT
          Choice% = 5
        CASE 5
          CALL GetRunningComments
          Choice% = 6
        CASE 6
          EXIT SUB
      END SELECT
    ELSE
      REDIM Item$(5)
      Title$ = "REVIEW PREVIOUS DATA MENU (FAILED)"
      Item$(1) = "View Deflection vs Station Plot(s)"
      Item$(2) = "View Corr. Deflection vs Station Plot(s)"
      Item$(3) = "View Outlier Analysis Curves"
      Item$(4) = "Add to Running Comments"
      Item$(5) = "Return to Structural Analysis Menu"
      CALL BarMenu(Title$, Item$(), Choice%, 0)
      SELECT CASE Choice%
        CASE 1
          CALL GetUniformityCurveParameters
          Choice% = 2
        CASE 2
          CALL GetCorrectedDeflectionParameters
          Choice% = 3
        CASE 3
          CALL GetOutlierCurveParameters(ExitCode%)
          Choice% = 4
        CASE 4
          CALL GetRunningComments
          Choice% = 5
        CASE 5
          EXIT SUB
      END SELECT
    END IF
  LOOP
END SUB

SUB StatisticsMenu (ExitCode%) STATIC
  SCREEN 0: WIDTH 80, 25: CLS
  Choice% = 1
  CALL WriteOverallStatistics
  DO
    REDIM Item$(6)
    Title$ = "STATISTICAL ASSESSMENT MENU"
    Item$(1) = "View Section Statistics"
    Item$(2) = "View Deflection vs Station Plot(s)"
    Item$(3) = "Add to Running Comments"
    Item$(4) = "Go on to SUBSECTION Analysis"
```

```
    Item$(5) = "Select NEW Data File"
    Item$(6) = "QUIT Program"
    CALL BarMenu(Title$, Item$(), Choice%, 0)
    SELECT CASE Choice%
      CASE 1
        CALL ShowOverallStats
        Choice% = 2
      CASE 2
        CALL GetUniformityCurveParameters
        Choice% = 3
      CASE 3
        CALL GetRunningComments
        Choice% = 4
      CASE 4
        ExitCode% = 1
        EXIT DO
      CASE 5
        CALL WriteRunningComments
        CLOSE #2
        ExitCode% = -2
        EXIT SUB
      CASE 6
        CALL WriteRunningComments
        CLOSE #2
        CALL Quit
    END SELECT
  LOOP
END SUB

SUB StructuralMenu (ExitCode%) STATIC
  SCREEN 0: WIDTH 80, 25: CLS
  SummaryOfResults% = True%
  Choice% = 1
  CALL CalcCompositeModulus
  SELECT CASE MatlCode%(1)
    CASE 700 'ACC
      CALL ChevronInputs
      CALL ChevronAnalysis(AnalysisFailed%)
      IF AnalysisFailed% THEN
        SummaryOfResults2% = False%
      ELSE
        CALL CalcFlexStats
        SummaryOfResults2% = True%
      END IF
    CASE 730 'PCC
      SummaryOfResults2% = True%
      CALL CalcVolumetricK
      CALL CalcRigidThickness
      CALL CalcRigidStats
  END SELECT
  DO
    IF SummaryOfResults2% THEN
      REDIM Item$(9)
      Title$ = "STRUCTURAL ANALYSIS MENU"
      Item$(1) = "View Structural Analysis Results"
      Item$(2) = "View Structural Analysis Results - Test Pits"
      Item$(3) = "View Composite Modulus vs Radius Plot(s)"
      SELECT CASE MatlCode%(1)
        CASE 700
          Item$(4) = "View Equiv. SN vs Station Plot(s)"
          Item$(5) = "View Subgrade Modulus vs Station Plot"
        CASE 730
          Item$(4) = "View Equiv. Thickness vs Station Plot(s)"
          Item$(5) = "View Volumetric K vs Station Plot"
      END SELECT
      Item$(6) = "Add to Running Comments"
      Item$(7) = "Review PREVIOUS Plots"
      Item$(8) = "Select NEW Data File"
      Item$(9) = "QUIT Program"
      CALL BarMenu(Title$, Item$(), Choice%, 0)
      SELECT CASE Choice%
        CASE 1
          SELECT CASE MatlCode%(1)
```

```
          CASE 700
            CALL ShowOverallFlexStats
          CASE 730
            CALL ShowOverallRigStats
        END SELECT
        Choice% = 2
      CASE 2
        IF NumTPStations% > 0 THEN
          SELECT CASE MatlCode%(1)
            CASE 700
              CALL ShowTPFlexStats
            CASE 730
              CALL ShowTPRigidStats
          END SELECT
        ELSE
          REDIM PUText$(1)
          PUText$(1) = "No test pit data is present, and no structural analysis results to view..."
          CALL PopupError
        END IF
        Choice% = 3
      CASE 3
        CALL CompositeModulusParameters
        Choice% = 4
      CASE 4
        SELECT CASE MatlCode%(1)
          CASE 700
            CALL PlotFlexSNCurves
          CASE 730
            CALL PlotRigThickCurves
        END SELECT
        Choice% = 5
      CASE 5
        SELECT CASE MatlCode%(1)
          CASE 700
            CALL PlotEsgCurves
          CASE 730
            CALL PlotVolKCurves
        END SELECT
        Choice% = 6
      CASE 6
        CALL GetRunningComments
        Choice% = 7
      CASE 7
        CALL ReviewMenu
      CASE 8
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteStructuralStats
        CALL WriteRunningComments
        CLOSE #2
        ExitCode% = -2
        EXIT SUB
      CASE 9
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteStructuralStats
        CALL WriteRunningComments
        CLOSE #2
        CALL Quit
    END SELECT
  ELSE
    REDIM Item$(4)
    Title$ = "STRUCTURAL ANALYSIS MENU (FAILED)"
    Item$(1) = "Add to Running Comments"
    Item$(2) = "Review PREVIOUS Plots"
    Item$(3) = "Select NEW Data File"
    Item$(4) = "QUIT Program"
    CALL BarMenu(Title$, Item$(), Choice%, 0)
    SELECT CASE Choice%
      CASE 1
        CALL GetRunningComments
        Choice% = 2
      CASE 2
```

```
        CALL ReviewMenu
      CASE 3
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteStructuralStats
        CALL WriteRunningComments
        CLOSE #2
        ExitCode% = -2
        EXIT SUB
      CASE 4
        CALL WriteSubsectionStats
        CALL WriteOutlierMsgs
        CALL WriteStructuralStats
        CALL WriteRunningComments
        CLOSE #2
        CALL Quit
    END SELECT
  END IF
LOOP
END SUB

SUB SubsectionAnalysisMenu (ExitCode%) STATIC
  SCREEN 0: WIDTH 80, 25: CLS
  Choice% = 1
  UsefulCombo% = False%
  DO
    REDIM Item$(8)
    Title$ = "SUBSECTION ANALYSIS MENU"
    Item$(1) = "Specify Material Properties"
    Item$(2) = "View Corr. Deflection vs Station Plot(s)"
    Item$(3) = "Define Subsection Limits"
    Item$(4) = "View Subsection Statistics"
    Item$(5) = "Add to Running Comments"
    Item$(6) = "Go on to OUTLIER Analysis"
    Item$(7) = "Select NEW Data File"
    Item$(8) = "QUIT Program"
    CALL BarMenu(Title$, Item$(), Choice%, 0)
    SELECT CASE Choice%
      CASE 1
        CALL GetMaterialProps
        Choice% = 2
        IF (LocationList%(2) AND MatlCode%(1) = 730) OR (LocationList%(4) AND MatlCode%(1) = 700) THEN
          UsefulCombo% = True%
        ELSE
          UsefulCombo% = False%
        END IF
      CASE 2
        IF SetMaterials% THEN
          CALL GetCorrectedDeflectionParameters
          Choice% = 3
        ELSE
          REDIM PUText$(2)
          PUText$(1) = " Materials MUST be specified for all pavement"
          PUText$(2) = "layers before viewing corrected deflections..."
          CALL PopupError
          Choice% = 1
        END IF
      CASE 3
        IF SetMaterials% AND UsefulCombo% THEN
          DO
            CALL GetSubsectionBoundaries(ExitCode%)
            CALL CalcSubSectionAlpha(Recalc%)
          LOOP WHILE Recalc%
          CALL CalcSubSectionStats
          SetSubSections% = True%
          Choice% = 4
        ELSEIF SetMaterials% THEN
          REDIM PUText$(2)
          PUText$(1) = "Subsection limits may ONLY be defined based on test location 1 data"
          PUText$(2) = "for rigid pavements or test location 3 for flexible pavements..."
          CALL PopupError
          Choice% = 5
        ELSE
```

```
        REDIM PUText$(2)
        PUText$(1) = "Materials MUST be specified for all pavement layers"
        PUText$(2) = "       before selecting subsection limits..."
        CALL PopupError
        Choice% = 1
      END IF
    CASE 4
      IF SetSubSections% THEN
        CALL ShowSubSectionStats
        Choice% = 5
      ELSEIF SetMaterials% AND NOT UsefulCombo% THEN
        REDIM PUText$(2)
        PUText$(1) = "Subsection limits may ONLY be defined based on test location 1 data"
        PUText$(2) = "for rigid pavements or test location 3 for flexible pavements..."
        CALL PopupError
        Choice% = 5
      ELSE
        REDIM PUText$(1)
        PUText$(1) = "Subsection limits MUST be specified before viewing statistics..."
        CALL PopupError
        Choice% = 3
      END IF
    CASE 5
      CALL GetRunningComments
      Choice% = 6
    CASE 6
      IF SetSubSections% THEN
        ExitCode% = 1
        EXIT DO
      ELSEIF SetMaterials% AND NOT UsefulCombo% THEN
        REDIM PUText$(2)
        PUText$(1) = "Outlier analysis may ONLY be performed based on test location 1 data"
        PUText$(2) = "for rigid pavements or test location 3 for flexible pavements..."
        CALL PopupError
        Choice% = 7
      ELSEIF SetMaterials% THEN
        REDIM PUText$(1)
        PUText$(1) = "Subsection limits MUST be specified before proceeding..."
        CALL PopupError
        Choice% = 3
      ELSE
        REDIM PUText$(1)
        PUText$(1) = "Materials MUST be specified for all pavement layers before proceeding..."
        CALL PopupError
        Choice% = 1
      END IF
    CASE 7        'select new data file
      IF SetSubSections% THEN CALL WriteSubsectionStats
      CALL WriteRunningComments
      CLOSE #2
      ExitCode% = -2
      EXIT SUB
    CASE 8
      IF SetSubSections% THEN CALL WriteSubsectionStats
      CALL WriteRunningComments
      CLOSE #2
      CALL Quit
  END SELECT
LOOP
END SUB
```

```
DECLARE SUB PlotSetup (XMin!, XMax!, YMin!, YMax!, PlotXMin!, PlotXMax!, PlotYMin!, PlotYMax!, LX!, XScale!)
DECLARE SUB EchoFlexStats (EchoSub%, EchoHeight%)
DECLARE SUB EchoRigStats (EchoSub%, EchoHeight%)
DECLARE SUB PlotCompositeModulus (PlotStation%, StatusLine$, PlotYMax!)
DECLARE FUNCTION TPNormRigidDefl! (hx!, I%, J%)
DECLARE FUNCTION StandardDeviation! (SSum!, SSum2!, ItemCount%) 'do not erase
DECLARE FUNCTION QPROUND$ (Number!, Places%)      'do not erase
DECLARE FUNCTION Log10 (Value!)                   'do not erase
DECLARE FUNCTION NormRigidDefl (hx!, I%, J%)      'do not erase
'$INCLUDE: 'declare.inc'
'$INCLUDE: 'fdeclar1.inc'
'$INCLUDE: 'fdeclar2.inc'
'$INCLUDE: 'cmnblank.inc'
'$INCLUDE: 'fwdcheck.inc'

CONST True% = -1, False% = 0, MaxNumStations% = 50, MaxLocations% = 6
CONST MaxDeflectors% = 7, MaxNumPeaks% = 16, MaxHeights% = 4, MaxLayers% = 10
CONST MinPointsInSubSection% = 4, ThresholdAlpha = 95, MaxTimes% = 10
CONST Pi = 3.141593

SUB CalcCompositeModulus STATIC
' Calculate subgrade modulus based on mean outer deflector readings:
  REDIM CompositeModulus(NumStations%, MaxHeight%, NumDeflectors%)
  N2 = .45                              'subgrade poisson's ratio
  P2 = 1 - N2 ^ 2
  CF = .5 * N2 + .875
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        CompositeModulus(I%, J%, 1) = 2 * P2 * LoadFactor / (Pi * TLPR * MeanNormDefl(I%, J%, 1) * DeflFactor)
        FOR K% = 2 TO NumDeflectors%
          IF MeanNormDefl(I%, J%, 1) <> 0 THEN
            CompositeModulus(I%, J%, K%) = CF * P2 * LoadFactor / (Pi * Radial(K%) * MeanNormDefl(I%, J%, K%) * DeflFactor
* RadFactor)
          END IF
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
END SUB

SUB CalcFlexStats STATIC
' Calculate Flexible Pavement Statistics:
  REDIM BCEsgMean(NumSubSect%, MaxHeight%), BCEsgStDev(NumSubSect%, MaxHeight%), BCEsgCV(NumSubSect%, MaxHeight%)
  REDIM EqSNMean(NumSubSect%, MaxHeight%), EqSNStDev(NumSubSect%, MaxHeight%), EqSNCV(NumSubSect%, MaxHeight%)
  LastStation% = 0
  FOR I% = 1 TO NumSubSect%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(I%) + FirstStation% - 1
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        SumBCEsg = 0:   SumEqSN = 0
        SumSqBCEsg = 0: SumSqEqSN = 0
        FOR K% = FirstStation% TO LastStation%
          SumBCEsg = SumBCEsg + BackCalcEsg(K%, J%)
          SumSqBCEsg = SumSqBCEsg + BackCalcEsg(K%, J%) ^ 2
          SumEqSN = SumEqSN + EquivalentSN(K%, J%)
          SumSqEqSN = SumSqEqSN + EquivalentSN(K%, J%) ^ 2
        NEXT K%
        BCEsgMean(I%, J%) = SumBCEsg / StationCount%(I%)
        BCEsgStDev(I%, J%) = StandardDeviation(SumBCEsg, SumSqBCEsg, StationCount%(I%))
        BCEsgCV(I%, J%) = BCEsgStDev(I%, J%) / BCEsgMean(I%, J%) * 100
        EqSNMean(I%, J%) = SumEqSN / StationCount%(I%)
        EqSNStDev(I%, J%) = StandardDeviation(SumEqSN, SumSqEqSN, StationCount%(I%))
        EqSNCV(I%, J%) = EqSNStDev(I%, J%) / EqSNMean(I%, J%) * 100
      END IF
    NEXT J%
  NEXT I%
END SUB

SUB CalcRigidStats STATIC
' Calculate Rigid Pavement Statistics:
  REDIM VolKMean(NumSubSect%, MaxHeight%), VolKStDev(NumSubSect%, MaxHeight%), VolKCV(NumSubSect%, MaxHeight%)
```

```
  REDIM RigThickMean(NumSubSect%, MaxHeight%), RigThickStDev(NumSubSect%, MaxHeight%), RigThickCV(NumSubSect%, MaxHeight%)
  LastStation% = 0
  FOR I% = 1 TO NumSubSect%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(I%) + FirstStation% - 1
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        SumVolK = 0:   SumRigThick = 0
        SumSqVolK = 0: SumSqRigThick = 0
        FOR K% = FirstStation% TO LastStation%
          SumVolK = SumVolK + VolumetricK(K%, J%)
          SumSqVolK = SumSqVolK + VolumetricK(K%, J%) ^ 2
          SumRigThick = SumRigThick + RigidThickness(K%, J%)
          SumSqRigThick = SumSqRigThick + RigidThickness(K%, J%) ^ 2
        NEXT K%
        VolKMean(I%, J%) = SumVolK / StationCount%(I%)
        VolKStDev(I%, J%) = StandardDeviation(SumVolK, SumSqVolK, StationCount%(I%))
        VolKCV(I%, J%) = VolKStDev(I%, J%) / VolKMean(I%, J%) * 100
        RigThickMean(I%, J%) = SumRigThick / StationCount%(I%)
        RigThickStDev(I%, J%) = StandardDeviation(SumRigThick, SumSqRigThick, StationCount%(I%))
        RigThickCV(I%, J%) = RigThickStDev(I%, J%) / RigThickMean(I%, J%) * 100
      END IF
    NEXT J%
  NEXT I%
END SUB

SUB CalcRigidThickness STATIC
' For PCC pavements calculate the required thickness:
  REDIM RigidThickness(NumStations%, MaxHeight%)
  REDIM TPRigidThickness(NumTPStations%, MaxHeight%), Linearity%(3)
  ModTotal = 5000000! * Thickness(1) ^ 3
  FOR I% = 2 TO NumLayers%
    IF MatlCode%(I%) = 730 THEN
      ModTotal = ModTotal + 5000000! * Thickness(I%) ^ 3
      IF I% >= 3 THEN
        FOR J% = 2 TO I% - 1
          ModTotal = ModTotal + 4500000! * Thickness(J%) ^ 3   'assume AC mat'l
        NEXT J%
      END IF
    END IF
  NEXT I%
  AvgThick = (ModTotal / 5000000!) ^ .3333333
  Tolerance = 1E-08
  TotalFactor = DeflFactor / LoadFactor
  HiThick = AvgThick * 1.15         'present definition
  LoThick = AvgThick * .65
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Linearity%(1) = Linearity%(1) + 1
        DCheck = MeanNormDefl(I%, J%, 1) * TotalFactor
        hmin = 2:  Dmax = NormRigidDefl(hmin, I%, J%)
        hmax = 26: Dmin = NormRigidDefl(hmax, I%, J%)
        IF DCheck < Dmin OR DCheck > Dmax THEN
          REDIM PUText$(1)
          PUText$(1) = "Unable to analyze rigid data point at station" + STR$(Station%(I%)) + "..."
          CALL PopupError
        ELSE
          DO
            hmid = (hmax + hmin) / 2
            Dmid = NormRigidDefl(hmid, I%, J%)
            Check = ABS(DCheck - Dmid)
            IF Check < Tolerance THEN RigidThickness(I%, J%) = hmid: EXIT DO
            IF DCheck >= Dmin AND DCheck < Dmid THEN
              hmin = hmid: Dmax = Dmid
            ELSE
              hmax = hmid: Dmin = Dmid
            END IF
          LOOP
        END IF
      END IF
    NEXT J%
  NEXT I%
```

```
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Linearity%(1) = Linearity%(1) + 1
        DCheck = TPMeanNormDefl(I%, J%, 1) * TotalFactor
        hmin = 2:  Dmax = TPNormRigidDefl(hmin, I%, J%)
        hmax = 26: Dmin = TPNormRigidDefl(hmax, I%, J%)
        IF DCheck < Dmin OR DCheck > Dmax THEN
          REDIM PUText$(1)
          PUText$(1) = "Unable to analyze rigid data point at station" + STR$(Station%(I%)) + "..."
          CALL PopupError
        ELSE
          DO
            hmid = (hmax + hmin) / 2
            Dmid = TPNormRigidDefl(hmid, I%, J%)
            Check = ABS(DCheck - Dmid)
            IF Check < Tolerance THEN TPRigidThickness(I%, J%) = hmid: EXIT DO
            IF DCheck >= Dmin AND DCheck < Dmid THEN
              hmin = hmid: Dmax = Dmid
            ELSE
              hmax = hmid: Dmin = Dmid
            END IF
          LOOP
        END IF
      END IF
    NEXT J%
  NEXT I%
END SUB

SUB CalcStationsInSubsection STATIC
  aa% = 0
  FOR I% = 1 TO NumSubSect%
    IF I% = 1 THEN
      FirstStat% = 0
      LastStat% = SubSectEnd%(I%)
    ELSE
      FirstStat% = SubSectEnd%(I% - 1)
      LastStat% = SubSectEnd%(I%)
    END IF
    StationCount%(I%) = 0
    FOR J% = 1 TO NumStations%
      IF FirstStat% <= Station%(J%) AND LastStat% > Station%(J%) THEN
        StationCount%(I%) = StationCount%(I%) + 1
        aa% = aa% + 1
      END IF
    NEXT J%
  NEXT I%
  IF aa% < NumStations% THEN StationCount%(NumSubSect%) = StationCount%(NumSubSect%) + (NumStations% - aa%)
  CALL NormalColor
  FOR I% = 1 TO NumSubSect%
    LOCATE 5 + 2 * I%, 60: PRINT USING "##"; StationCount%(I%);
    IF StationCount%(I%) >= MinPointsInSubSection% THEN
      PRINT "         ok  "
    ELSE
      CALL HiliteColor
      PRINT "       NOT ok"
      CALL NormalColor
    END IF
  NEXT I%
END SUB

SUB CalcVolumetricK STATIC
' For PCC pavements calculate the volumetric modulus of subgrade reaction, k:
  REDIM VolumetricK(NumStations%, MaxHeight%), TPVolumetricK(NumTPStations%, MaxHeight%)
  TotalFactor = LoadFactor / (DeflFactor * RadFactor ^ 2)
  P2 = 2 * Pi
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        IF MeanNormDefl(I%, J%, NumDeflectors%) < MeanNormDefl(I%, J%, NumDeflectors% - 1) THEN
          Radial(NumDeflectors% + 1) = (MeanNormDefl(I%, J%, NumDeflectors%) / ((MeanNormDefl(I%, J%, NumDeflectors% - 1)
- MeanNormDefl(I%, J%, NumDeflectors%)) / (Radial(NumDeflectors%) - Radial(NumDeflectors% - 1)))) + Radial(NumDeflectors%)
        IF English% THEN
```

```
          IF Radial(NumDeflectors% + 1) > 78 THEN Radial(NumDeflectors% + 1) = 78
        ELSE
          IF Radial(NumDeflectors% + 1) > 198 THEN Radial(NumDeflectors% + 1) = 198
        END IF
      ELSE
        IF English% THEN
          Radial(NumDeflectors% + 1) = 78
        ELSE
          Radial(NumDeflectors% + 1) = 198
        END IF
      END IF
      MeanNormDefl(I%, J%, NumDeflectors% + 1) = 0
      VolT = 0: VolR = 0
      FOR K% = 1 TO NumDeflectors%
        RadialBase = Radial(K% + 1) - Radial(K%)
        IF MeanNormDefl(I%, J%, K%) > MeanNormDefl(I%, J%, K% + 1) THEN
          Factor1 = 1: Factor2 = 1
        ELSE
          Factor1 = 2: Factor2 = 0
        END IF
        CentroidR = Radial(K%) + RadialBase / 2
        CentroidT = Radial(K%) + Factor1 * RadialBase / 3
        IF K% <> NumDeflectors% THEN
          AreaR = RadialBase * MeanNormDefl(I%, J%, K% + Factor2)
          VolR = VolR + AreaR * CentroidR * P2
        END IF
        AreaT = .5 * RadialBase * ABS(MeanNormDefl(I%, J%, K%) - MeanNormDefl(I%, J%, K% + 1))
        VolT = VolT + AreaT * CentroidT * P2
      NEXT K%
      VolumetricK(I%, J%) = TotalFactor / (VolT + VolR)^pci
    END IF
  NEXT J%
NEXT I%
FOR I% = 1 TO NumTPStations%
  FOR J% = 1 TO MaxHeight%
    IF HeightList%(J%) THEN
      IF TPMeanNormDefl(I%, J%, NumDeflectors%) < TPMeanNormDefl(I%, J%, NumDeflectors% - 1) THEN
        Radial(NumDeflectors% + 1) = (TPMeanNormDefl(I%, J%, NumDeflectors%) / ((TPMeanNormDefl(I%, J%, NumDeflectors%
- 1) - TPMeanNormDefl(I%, J%, NumDeflectors%)) / (Radial(NumDeflectors%) - Radial(NumDeflectors% - 1)))) +
Radial(NumDeflectors% _
)
        IF English% THEN
          IF Radial(NumDeflectors% + 1) > 78 THEN Radial(NumDeflectors% + 1) = 78
        ELSE
          IF Radial(NumDeflectors% + 1) > 198 THEN Radial(NumDeflectors% + 1) = 198
        END IF
      ELSE
        IF English% THEN
          Radial(NumDeflectors% + 1) = 78
        ELSE
          Radial(NumDeflectors% + 1) = 198
        END IF
      END IF
      TPMeanNormDefl(I%, J%, NumDeflectors% + 1) = 0
      VolT = 0: VolR = 0
      FOR K% = 1 TO NumDeflectors%
        RadialBase = Radial(K% + 1) - Radial(K%)
        IF TPMeanNormDefl(I%, J%, K%) > TPMeanNormDefl(I%, J%, K% + 1) THEN
          Factor1 = 1: Factor2 = 1
        ELSE
          Factor1 = 2: Factor2 = 0
        END IF
        CentroidR = Radial(K%) + RadialBase / 2
        CentroidT = Radial(K%) + Factor1 * RadialBase / 3
        IF K% <> NumDeflectors% THEN
          AreaR = RadialBase * TPMeanNormDefl(I%, J%, K% + Factor2)
          VolR = VolR + AreaR * CentroidR * P2
        END IF
        AreaT = .5 * RadialBase * ABS(TPMeanNormDefl(I%, J%, K%) - TPMeanNormDefl(I%, J%, K% + 1))
        VolT = VolT + AreaT * CentroidT * P2
      NEXT K%
      TPVolumetricK(I%, J%) = TotalFactor / (VolT + VolR)^pci
    END IF
```

```
    NEXT J%
  NEXT I%
END SUB

SUB ChevronInputs STATIC
  SCREEN 0: WIDTH 80, 25: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  Flexible Pavement Structural Analysis  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL NormalColor
  LOCATE 5, 8: PRINT "Depth to Rigid Foundation:         feet (Enter zero if not present)"
  LOCATE 25, 4
  PRINT "                                          PgDn";
  TotalThick = 0
  FOR I% = 1 TO NumLayers%
    TotalThick = TotalThick + Thickness(I%)       'inches
  NEXT I%
  TotalThick = TotalThick / 12                    'feet
  TT$ = QPTrim$(DefaultReal$(TotalThick, "R", 4, "##.#"))
  DepthRigidLayer = 0
  DO
    OldDepthRigidLayer = DepthRigidLayer
    CALL GetReal(5, 36, 5, "###.#", DepthRigidLayer, "L", 0, 0, "", Dummy%, ExitCode%)
    IF DepthRigidLayer <> 0 AND (DepthRigidLayer < TotalThick OR DepthRigidLayer > 100) THEN
      REDIM PUText$(2)
      PUText$(1) = "Depth to a rigid foundation must be specified as either 0 (no"
      PUText$(2) = "known rigid layer) or between " + TT$ + " and 100 feet ONLY..."
      CALL PopupError
      DepthRigidLayer = OldDepthRigidLayer
      ExitCode% = 0
    END IF
    SELECT CASE ExitCode%                         'determine next action
      CASE 81           'PgDn
        IF DepthRigidLayer = 0 THEN DepthRigidLayer = 100
        DepthRigidLayer = DepthRigidLayer * 12
        CALL NormalColor
        LOCATE 5, 2: PRINT SPACE$(78);
        ExitCode% = 1
        EXIT DO
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB CompositeModulusParameters STATIC
  REDIM Y(NumDeflectors% * MaxHeight% * NumStations%)
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        FOR K% = 1 TO NumDeflectors%
          Count% = Count% + 1
          Y(Count%) = CompositeModulus(I%, J%, K%)
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  T1 = INT(LOG(PlotYMax) / LOG(10))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  StatusLine$ = "        F10:ExitPlots  Home  End  PgUp  PgDn "
  PlotStation% = 1
  DoPlot% = True%
  DO
    IF DoPlot% THEN CALL PlotCompositeModulus(PlotStation%, StatusLine$, PlotYMax)
    DO
```

```
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    IF NOT DoPlot% THEN LOCATE 32, 1: PRINT SPACE$(13)
    DoPlot% = True%
    SELECT CASE KCode%
      CASE 71           'home
        PlotStation% = 1
      CASE 79           'end
        PlotStation% = NumStations%
      CASE 73           'PgUp
        IF PlotStation% > 1 THEN
          PlotStation% = PlotStation% - 1
        ELSE
          CALL Bonk
          COLOR 12: LOCATE 32, 1: PRINT "First Station"
          DoPlot% = False%
        END IF
      CASE 81           'PgDn
        IF PlotStation% < NumStations% THEN
          PlotStation% = PlotStation% + 1
        ELSE
          CALL Bonk
          COLOR 12: LOCATE 32, 1: PRINT "Last Station"
          DoPlot% = False%
        END IF
      CASE 68           'F10:Exit Plots
        ExitCode% = 0
        EXIT SUB
      CASE 60           'F2:Screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB DisplayCopyright STATIC
  SCREEN 0: WIDTH 80: CLS
  PRINT
  PRINT "
  PRINT "
  PRINT "
  PRINT "
  PRINT "
  LOCATE 10, 28: PRINT "FWD Data Checking Software"
  LOCATE 12, 35: PRINT "Version 2.10"
  LOCATE 14, 20: PRINT "         For EXCLUSIVE Use by the"
  LOCATE 15, 20: PRINT "Strategic Highway Research Program  (SHRP)"
  LOCATE 16, 20: PRINT " and its contractors and sub-contractors."
  LOCATE 22, 5:  PRINT "Support material Copyright (c) 1989 - 1992      PCS/Law Engineering Inc."
  LOCATE 23, 12: PRINT "Additional material Copyright (c) 1988   Crescent Software"
  SLEEP 5
  CALL ClearBuf
END SUB

SUB EchoFlexStats (EchoSub%, EchoHeight%) STATIC
  SCREEN 0: WIDTH 80, 50: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  FLEXIBLE Pavement Thickness Statistics  "
  LOCATE 2, 41 - LEN(Title$) / 2: PRINT Title$
  CALL NormalColor
  StatusLine$ = "                                        PgUp  PgDn"
  LOCATE 50, 4: PRINT StatusLine$;
  LOCATE 4, 5: PRINT "Data for section"
  LOCATE 5, 5: PRINT "      Subsection"
  LOCATE 6, 5: PRINT "     Drop height"
  CALL HiliteColor
  LOCATE 4, 22: PRINT LEFT$(File$, 7)
  LOCATE 5, 22: PRINT USING "#"; EchoSub%
  LOCATE 6, 22: PRINT USING "#"; EchoHeight%
```

```
  CALL NormalColor
  Header1$ = "                    Subgrade        Effective"
  Header2$ = "  Station           Modulus           SN"
  Header3$ = "------------      ------------     ----------"
  Format1$ = "    ###             ######           ##.##"
  Footer1$ = "      Overall Mean: ######           ##.##"
  Footer2$ = "Standard Deviation: ######           ##.##"
  Footer3$ = "Coeff Of Variation: ###.##%         ###.##%"
  Col% = 41 - LEN(Header1$) / 2
  LOCATE 8, Col%: PRINT Header1$
  LOCATE , Col%: PRINT Header2$
  LOCATE , Col%: PRINT Header3$
  LastStation% = 0
  FOR I% = 1 TO EchoSub%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(I%) + FirstStation% - 1
  NEXT I%
  FOR I% = FirstStation% TO LastStation%
    LOCATE , Col%: PRINT USING Format1$; Station%(I%); BackCalcEsg(I%, EchoHeight%); EquivalentSN(I%, EchoHeight%)
  NEXT I%
  LOCATE , Col%: PRINT Header3$
  LOCATE , Col%: PRINT USING Footer1$; BCEsgMean(EchoSub%, EchoHeight%); EqSNMean(EchoSub%, EchoHeight%)
  LOCATE , Col%: PRINT USING Footer2$; BCEsgStDev(EchoSub%, EchoHeight%); EqSNStDev(EchoSub%, EchoHeight%)
  LOCATE , Col%: PRINT USING Footer3$; BCEsgCV(EchoSub%, EchoHeight%); EqSNCV(EchoSub%, EchoHeight%)
END SUB

SUB EchoRigStats (EchoSub%, EchoHeight%) STATIC
  SCREEN 0: WIDTH 80, 50: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  RIGID Pavement Thickness Statistics  "
  LOCATE 2, 41 - LEN(Title$) / 2: PRINT Title$
  CALL NormalColor
  StatusLine$ = "                                        PgUp  PgDn"
  LOCATE 50, 4: PRINT StatusLine$;
  LOCATE 4, 5: PRINT "Data for section "
  LOCATE 5, 5: PRINT "      Subsection "
  LOCATE 6, 5: PRINT "     Drop height "
  CALL HiliteColor
  LOCATE 4, 22: PRINT LEFT$(File$, 7)
  LOCATE 5, 22: PRINT USING "#"; EchoSub%
  LOCATE 6, 22: PRINT USING "#"; EchoHeight%
  CALL NormalColor
  Header1$ = "                 Volumetric        Effective"
  Header2$ = "  Station            k             Thickness"
  Header3$ = "------------     ------------      ----------"
  Format1$ = "    ###              ####             ##.##"
  Footer1$ = "      Overall Mean: ####             ##.##"
  Footer2$ = "Standard Deviation: ####             ##.##"
  Footer3$ = "Coeff Of Variation: ###.##%         ###.##%"
  Col% = 41 - LEN(Header1$) / 2
  LOCATE 8, Col%: PRINT Header1$
  LOCATE , Col%: PRINT Header2$
  LOCATE , Col%: PRINT Header3$
  LastStation% = 0
  FOR I% = 1 TO EchoSub%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(I%) + FirstStation% - 1
  NEXT I%
  FOR I% = FirstStation% TO LastStation%
    LOCATE , Col%: PRINT USING Format1$; Station%(I%); VolumetricK(I%, EchoHeight%); RigidThickness(I%, EchoHeight%)
  NEXT I%
  LOCATE , Col%: PRINT Header3$
  LOCATE , Col%: PRINT USING Footer1$; VolKMean(EchoSub%, EchoHeight%); RigThickMean(EchoSub%, EchoHeight%)
  LOCATE , Col%: PRINT USING Footer2$; VolKStDev(EchoSub%, EchoHeight%); RigThickStDev(EchoSub%, EchoHeight%)
  LOCATE , Col%: PRINT USING Footer3$; VolKCV(EchoSub%, EchoHeight%); RigThickCV(EchoSub%, EchoHeight%)
END SUB

SUB LinearLeastSquaresFit (Num%, X(), Y(), Constant, Slope, Rsqd) STATIC
  SumX = 0: SumXSq = 0: SumY = 0: SumYSq = 0: SumXY = 0
  FOR aaa% = 1 TO Num%
    SumX = SumX + X(aaa%)
    SumXSq = SumXSq + X(aaa%) ^ 2
```

```
    SumY = SumY + Y(aaa%)
    SumYSq = SumYSq + Y(aaa%) ^ 2
    SumXY = SumXY + X(aaa%) * Y(aaa%)
  NEXT aaa%
  T1 = (Num% * SumXSq) - (SumX ^ 2)
  T2 = (Num% * SumYSq) - (SumY ^ 2)
  T3 = (Num% * SumXY) - (SumX * SumY)
  Constant = ((SumXSq * SumY) - (SumX * SumXY)) / T1
  Slope = T3 / T1
  Rsqd = T3 ^ 2 / (T1 * T2)
END SUB

FUNCTION Log10 (Value) STATIC
  Log10 = LOG(Value) / LOG(10)
END FUNCTION

FUNCTION NormRigidDefl (hx, I%, J%) STATIC
' Westergaard solution:
  Gamma# = .5772156649#  'Euler's constant
  L = ((1000 * Modulus(1) * hx ^ 3) / (12 * (1 - .15 ^ 2) * VolumetricK(I%, J%))) ^ (1 / 4)
  C1 = 1 + (1 / (2 * Pi)) * (LOG(TLPR / (2 * L)) + Gamma# - 1.25) * (TLPR / L) ^ 2
  NormRigidDefl = 1 / (8 * VolumetricK(I%, J%) * L ^ 2) * C1
END FUNCTION

SUB PlotCompositeModulus (PlotStation%, StatusLine$, PlotYMax) STATIC
  SCREEN 12: WIDTH 80, 60: COLOR 0: CLS
  XMin = 130: XMax = 630
  YMin = 20: YMax = 420
  LINE (XMin, YMin)-(XMax, YMax), 6, B
  LX = 500: PlotXMax = Radial(NumDeflectors%): PlotXMin = Radial(1)
  XScale = ABS(PlotXMax - PlotXMin) / LX
  FOR M% = 2 TO NumDeflectors% - 1 STEP 1          'radial distance
    PM = XMin + (Radial(M%) - PlotXMin) / XScale
    LINE (PM, YMin)-(PM, YMax), 7, , &HCCCC
  NEXT M%
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 24: PRINT "Composite Modulus vs Deflector for Section: "; PS$
  LOCATE 26, 1: PRINT "Composite"
  LOCATE 27, 1: PRINT " Modulus "
  LOCATE 28, 1: PRINT "   Ec    "
  IF English% THEN Format$ = "###.#" ELSE Format$ = "#####"
  FOR M% = 1 TO NumDeflectors% STEP NumDeflectors% - 1
    Col% = 16 + (Radial(M%) - Radial(1)) / (Radial(NumDeflectors%) - Radial(1)) * 60
    LOCATE 54, Col%: PRINT USING Format$; Radial(M%);
  NEXT M%
  LOCATE 56, 42: PRINT "Radial Distance"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  COLOR 14
  LOCATE 30, 1: PRINT USING "Station ###"; Station%(PlotStation%)
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "#.###^^^^"
  COLOR 15
  LOCATE 53, 8:  PRINT USING FA$; PlotYMin;
  LOCATE 3, 8:   PRINT USING FA$; PlotYMax
  LOCATE 58, 56: PRINT "Drop Height";
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN
      SELECT CASE PlotHeight%
        CASE 1
          LineColor% = 1
        CASE 2
          LineColor% = 2
        CASE 3
          LineColor% = 3
        CASE 4
```

```
        LineColor% = 4
      END SELECT
      Started% = False%
      FOR J% = 1 TO NumDeflectors%
        PX = X0 + (Radial(J%) - PlotXMin) / XScale
        PY = Y0 - (CompositeModulus(PlotStation%, PlotHeight%, J%) - PlotYMin) / YScale
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      NEXT J%
      COLOR LineColor%
      PRINT QPRTrim$(STR$(PlotHeight%));
      IF PlotHeight% < MaxHeight% THEN COLOR 15: PRINT ",";
    END IF
  NEXT PlotHeight%
END SUB

SUB PlotEsgCurves STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 28: PRINT "Subgrade Elastic Modulus for Section: "; PS$
  LOCATE 27, 1: PRINT " Subgrade "
  LOCATE 28, 1: PRINT " Elastic  "
  LOCATE 29, 1: PRINT " Modulus  "
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F10:ExitPlots"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%))
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = BackCalcEsg(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = TPBackCalcEsg(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  T1 = INT(Log10(PlotYMax))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "#.###^^^^"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 56: PRINT "Drop Height";
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN
      SELECT CASE PlotHeight%
        CASE 1
```

```
        LineColor% = 1
      CASE 2
        LineColor% = 2
      CASE 3
        LineColor% = 3
      CASE 4
        LineColor% = 4
    END SELECT
    Started% = False%
    FOR I% = 1 TO NumStations%
      PX = X0 + (Station%(I%) - PlotXMin) / XScale
      PY = Y0 - (BackCalcEsg(I%, PlotHeight%) - PlotYMin) / YScale
      IF Started% THEN LINE -(PX, PY), LineColor%
      CIRCLE (PX, PY), 3, LineColor%
      Started% = True%
    NEXT I%
    FOR I% = 1 TO NumTPStations%
      PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
      PY = Y0 - (TPBackCalcEsg(I%, PlotHeight%) - PlotYMin) / YScale
      CIRCLE (PX, PY), 3, LineColor%
      PAINT (PX, PY), LineColor%, LineColor%
    NEXT I%
    COLOR LineColor%
    PRINT QPRTrim$(STR$(PlotHeight%));
    IF PlotHeight% < MaxHeight% THEN COLOR 15: PRINT ",";
  END IF
NEXT PlotHeight%
DO
  DO
    sdr$ = INKEY$
  LOOP WHILE sdr$ = ""
  KCode% = KeyCode%(sdr$)
  SELECT CASE KCode%
    CASE 60       'F2:screen dump
      CALL ScrnDump(ScrnDumpRes$, 1, 0)
      LPRINT CHR$(12);
    CASE 68       'F10:ExitPlots
      EXIT DO
    CASE ELSE
      'do nothing
  END SELECT
LOOP
END SUB

SUB PlotFlexSNCurves STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 25: PRINT "Equivalent Structural Number for Section: "; PS$
  LOCATE 27, 1: PRINT "Structural"
  LOCATE 28, 1: PRINT "  Number  "
  LOCATE 29, 1: PRINT "   (SN)   "
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F10:ExitPlots"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%))
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = EquivalentSN(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
```

```
        Count% = Count% + 1
        Y(Count%) = TPEquivalentSN(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  IF PlotYMax < HiSN THEN PlotYMax = HiSN
  T1 = INT(Log10(PlotYMax))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
' Limits based on lo and hi ai values
  PY = Y0 - (HiSN - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14, , &HFF00
  PY = Y0 - (LoSN - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14, , &HFF00
  PY = Y0 - (AvgSN - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14
  FA$ = "  ###.##"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 56: PRINT "Drop Height";
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN
      SELECT CASE PlotHeight%
        CASE 1
          LineColor% = 1
        CASE 2
          LineColor% = 2
        CASE 3
          LineColor% = 3
        CASE 4
          LineColor% = 4
      END SELECT
      Started% = False%
      FOR I% = 1 TO NumStations%
        PX = X0 + (Station%(I%) - PlotXMin) / XScale
        PY = Y0 - (EquivalentSN(I%, PlotHeight%) - PlotYMin) / YScale
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      NEXT I%
      FOR I% = 1 TO NumTPStations%
        PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
        PY = Y0 - (TPEquivalentSN(I%, PlotHeight%) - PlotYMin) / YScale
        CIRCLE (PX, PY), 3, LineColor%
        PAINT (PX, PY), LineColor%, LineColor%
      NEXT I%
      COLOR LineColor%
      PRINT QPRTrim$(STR$(PlotHeight%));
      IF PlotHeight% < MaxHeight% THEN COLOR 15: PRINT ",";
    END IF
  NEXT PlotHeight%
  DO
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 60       'F2:screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
      CASE 68       'F10:ExitPlots
        EXIT DO
```

```
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB PlotRigThickCurves STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 21: PRINT "Westergaard based Rigid Thickness for Section: "; PS$
  LOCATE 26, 1: PRINT "Effective"
  LOCATE 27, 1: PRINT "  Rigid  "
  LOCATE 28, 1: PRINT " Pavement"
  LOCATE 29, 1: PRINT "Thickness"
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F10:ExitPlots"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%))
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = RigidThickness(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = TPRigidThickness(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxS!(SEG Y(1), Count%)
  ERASE Y
  IF PlotYMax < HiThick THEN PlotYMax = HiThick
  T1 = INT(Log10(PlotYMax))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
' Limits based on actual Thickness (1.15 h to 0.65 h)
  PY = Y0 - (HiThick - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14, , &HFF00
  PY = Y0 - (LoThick - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14, , &HFF00
' Given Pavement Thickness, h
  PY = Y0 - (AvgThick - PlotYMin) / YScale
  LINE (X0, PY)-(X0 + LX, PY), 14
  FA$ = "  ###.##"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 56: PRINT "Drop Height";
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN
      SELECT CASE PlotHeight%
        CASE 1
          LineColor% = 1
```

```
        CASE 2
          LineColor% = 2
        CASE 3
          LineColor% = 3
        CASE 4
          LineColor% = 4
      END SELECT
      Started% = False%
      FOR I% = 1 TO NumStations%
        PX = X0 + (Station%(I%) - PlotXMin) / XScale
        PY = Y0 - (RigidThickness(I%, PlotHeight%) - PlotYMin) / YScale
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      NEXT I%
      FOR I% = 1 TO NumTPStations%
        PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
        PY = Y0 - (TPRigidThickness(I%, PlotHeight%) - PlotYMin) / YScale
        CIRCLE (PX, PY), 3, LineColor%
        PAINT (PX, PY), LineColor%, LineColor%
      NEXT I%
      COLOR LineColor%
      PRINT QPRTrim$(STR$(PlotHeight%));
      IF PlotHeight% < MaxHeight% THEN COLOR 15: PRINT ",";
    END IF
  NEXT PlotHeight%
  DO
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 60       'F2:screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
      CASE 68       'F10:ExitPlots
        EXIT DO
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB PlotVolKCurves STATIC
  CALL PlotSetup(XMin, XMax, YMin, YMax, PlotXMin, PlotXMax, PlotYMin, PlotYMax, LX, XScale)
  COLOR 15
  PS$ = LEFT$(File$, 7)
  LOCATE 1, 18: PRINT "Volumetric Modulus of Subgrade Reaction for Section: "; PS$
  LOCATE 26, 1: PRINT "Volumetric"
  LOCATE 27, 1: PRINT "Modulus of"
  LOCATE 28, 1: PRINT " Subgrade"
  LOCATE 29, 1: PRINT " Reaction"
  LOCATE 30, 1: PRINT "   (k)"
  FOR Value% = -100 TO 600 STEP 100
    Col% = 14 + (1 + Value% / 100) * 9
    LOCATE 54, Col%: PRINT USING "####"; Value%
  NEXT Value%
  LOCATE 56, 42: PRINT "Station (ft)"
  StatusLine$ = "F10:ExitPlots"
  COLOR 7
  LOCATE 60, 4: PRINT StatusLine$;
  REDIM Y(MaxHeight% * (NumStations% + NumTPStations%))
  Count% = 0
  FOR I% = 1 TO NumStations%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = VolumetricK(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  FOR I% = 1 TO NumTPStations%
    FOR J% = 1 TO MaxHeight%
```

```
      IF HeightList%(J%) THEN
        Count% = Count% + 1
        Y(Count%) = TPVolumetricK(I%, J%)
      END IF
    NEXT J%
  NEXT I%
  PlotYMax = MaxSI(SEG Y(1), Count%)
  ERASE Y
  T1 = INT(Log10(PlotYMax))
  T2 = 10 ^ T1
  T3 = FIX(PlotYMax / T2) + 1
  PlotYMax = T3 * T2
  PlotYMin = 0
  LY = 400: Y0 = 420: X0 = XMin
  YScale = ABS(PlotYMax - PlotYMin) / LY
  FOR M = 1 TO 3                'tic marks
    PM = Y0 - M * LY / 4
    LINE (X0, PM)-(X0 + LX, PM), 7, , &HCCCC
  NEXT M
  FA$ = "#.###^^^^"
  COLOR 15
  LOCATE 53, 8: PRINT USING FA$; PlotYMin;
  LOCATE 3, 8: PRINT USING FA$; PlotYMax
  LOCATE 58, 56: PRINT "Drop Height";
  FOR PlotHeight% = 1 TO MaxHeight%
    IF HeightList%(PlotHeight%) THEN
      SELECT CASE PlotHeight%
        CASE 1
          LineColor% = 1
        CASE 2
          LineColor% = 2
        CASE 3
          LineColor% = 3
        CASE 4
          LineColor% = 4
      END SELECT
      Started% = False%
      FOR I% = 1 TO NumStations%
        PX = X0 + (Station%(I%) - PlotXMin) / XScale
        PY = Y0 - (VolumetricK(I%, PlotHeight%) - PlotYMin) / YScale
        IF Started% THEN LINE -(PX, PY), LineColor%
        CIRCLE (PX, PY), 3, LineColor%
        Started% = True%
      NEXT I%
      FOR I% = 1 TO NumTPStations%
        PX = X0 + (TPStation%(I%) - PlotXMin) / XScale
        PY = Y0 - (TPVolumetricK(I%, PlotHeight%) - PlotYMin) / YScale
        CIRCLE (PX, PY), 3, LineColor%
        PAINT (PX, PY), LineColor%, LineColor%
      NEXT I%
      COLOR LineColor%
      PRINT QPRTrim$(STR$(PlotHeight%));
      IF PlotHeight% < MaxHeight% THEN COLOR 15: PRINT ",";
    END IF
  NEXT PlotHeight%
  DO
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 60       'F2:screen dump
        CALL ScrnDump(ScrnDumpRes$, 1, 0)
        LPRINT CHR$(12);
      CASE 68       'F10:ExitPlots
        EXIT DO
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

FUNCTION QPROUND$ (Amount!, Places%) STATIC
```

```
'This function accepts a double precision value and a number of decimal
'places to round it to, and returns the answer in a string.  The number
'must be processed as a string because many double precision values can
'not be represented accurately any other way.  To prove this, enter the
'following line into the QuickBASIC 4 editor and watch what happens:
'
'    X# = .0691#
'
'External routines required:  ASCII%, FUsing
  Amount$ = STR$(Amount!)                       'first convert into a string
  Lead% = 1                                     'allow one leading digit
  DO WHILE ABS(Amount!) + .000001# >= 10# ^ Lead%
    Lead% = Lead% + 1                           'get number of leading digits
  LOOP                                          'the .000001# avoids a QB bug w/80x87
  Dec$ = MID$(".", ABS(Places% = 0) + 1)        'make a "." if needed
  Plus$ = MID$("+", SGN(Amount!) + 2)           'make a "+" if needed
  Mask$ = Plus$ + STRING$(Lead%, "#") + Dec$ + STRING$(Places%, "#")
  Mask$ = FUsing$(Amount$, Mask$)               'let Chris do the dirty work
  IF ASCII%(Mask$) = 48 THEN                    'if there's a leading zero
    Mask$ = MID$(Mask$, 2)                      'strip it
  END IF
  QPROUND$ = Mask$                              'assign the function
END FUNCTION

SUB Quit STATIC
  SCREEN 0: WIDTH 80, 25: COLOR 7, 0, 0: CLS
  PRINT "Thank you for using FWDCHECK.  Have a nice day!"
  END
END SUB

SUB ShowOverallFlexStats STATIC
  EchoSub% = 1
  FOR J% = 1 TO MaxHeight%
    IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
  NEXT J%
  DO
    CALL EchoFlexStats(EchoSub%, EchoHeight%)
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 73       'PgUp
        NewEchoHeight% = 0
        FOR J% = EchoHeight% - 1 TO 1 STEP -1
          IF HeightList%(J%) THEN NewEchoHeight% = J%: EXIT FOR
        NEXT J%
        IF NewEchoHeight% <> 0 THEN
          EchoHeight% = NewEchoHeight%
        ELSEIF NewEchoHeight% = 0 AND EchoSub% > 1 THEN
          EchoSub% = EchoSub% - 1
          FOR J% = MaxHeight% TO 1 STEP -1
            IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
          NEXT J%
        ELSE
          WIDTH 80, 25
          EXIT DO
        END IF
      CASE 81       'PgDn
        NewEchoHeight% = 0
        FOR J% = EchoHeight% + 1 TO MaxHeight%
          IF HeightList%(J%) THEN NewEchoHeight% = J%: EXIT FOR
        NEXT J%
        IF NewEchoHeight% <> 0 THEN
          EchoHeight% = NewEchoHeight%
        ELSEIF NewEchoHeight% = 0 AND EchoSub% < NumSubSect% THEN
          EchoSub% = EchoSub% + 1
          FOR J% = 1 TO MaxHeight%
            IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
          NEXT J%
        ELSE
          WIDTH 80, 25
          EXIT DO
```

```
        END IF
      CASE 60       'F2:PrtSc
        CALL PrtSc0(1)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB ShowOverallRigStats STATIC
  EchoSub% = 1
  FOR J% = 1 TO MaxHeight%
    IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
  NEXT J%
  DO
    CALL EchoRigStats(EchoSub%, EchoHeight%)
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 73       'PgUp
        NewEchoHeight% = 0
        FOR J% = EchoHeight% - 1 TO 1 STEP -1
          IF HeightList%(J%) THEN NewEchoHeight% = J%: EXIT FOR
        NEXT J%
        IF NewEchoHeight% <> 0 THEN
          EchoHeight% = NewEchoHeight%
        ELSEIF NewEchoHeight% = 0 AND EchoSub% > 1 THEN
          EchoSub% = EchoSub% - 1
          FOR J% = MaxHeight% TO 1 STEP -1
            IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
          NEXT J%
        ELSE
          WIDTH 80, 25
          EXIT DO
        END IF
      CASE 81       'PgDn
        NewEchoHeight% = 0
        FOR J% = EchoHeight% + 1 TO MaxHeight%
          IF HeightList%(J%) THEN NewEchoHeight% = J%: EXIT FOR
        NEXT J%
        IF NewEchoHeight% <> 0 THEN
          EchoHeight% = NewEchoHeight%
        ELSEIF NewEchoHeight% = 0 AND EchoSub% < NumSubSect% THEN
          EchoSub% = EchoSub% + 1
          FOR J% = 1 TO MaxHeight%
            IF HeightList%(J%) THEN EchoHeight% = J%: EXIT FOR
          NEXT J%
        ELSE
          WIDTH 80, 25
          EXIT DO
        END IF
      CASE 60       'F2:PrtSc
        CALL PrtSc0(1)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB ShowOverallStats STATIC
  SCREEN 0: WIDTH 80, 50: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  UNCORRECTED Deflection Statistics  "
  TL% = LEN(Title$)
  Col% = ((80 - TL%) / 2) + 1
  LOCATE 2, Col%: PRINT Title$
  CALL HiliteColor
  LOCATE 4, 22: PRINT LEFT$(File$, 7)
```

```
  CALL NormalColor
  StatusLine$ = "                                                PgDn"
  LOCATE 50, 4: PRINT StatusLine$;
  LOCATE 4, 5: PRINT "Data for section "
  IF English% THEN ZText$ = " (mils/kip)" ELSE ZText$ = " (microns/kPa)"
  LOCATE 4, 35: PRINT "Mean  Values"; ZText$
  Header1$ = "Test  Drop   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor"
  Header2$ = "Loc.   Ht       1        2        3        4        5        6        7"
  Header3$ = "----  ----   ------   ------   ------   ------   ------   ------   ------"
  LOCATE 6, 4: PRINT Header1$
  LOCATE 7, 4: PRINT Header2$
  LOCATE 8, 4: PRINT Header3$
  Format1$ = "    #     # "
  Format2$ = "          # "
  IF English% THEN
    FormatA$ = "  ##.#####"
  ELSE
    FormatA$ = "  #.#####"
  END IF
  FOR I% = 1 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          LOCATE , 2
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT USING Format1$; I% - 1; J%;
          ELSE
            PRINT USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            IF I% = 1 THEN
              PRINT USING FormatA$; TPOverallMeanNormDefl(J%, K%) * 1000;
            ELSE
              PRINT USING FormatA$; OverallMeanNormDefl(I%, J%, K%) * 1000;
            END IF
          NEXT K%
          PRINT
        END IF
      NEXT J%
      PRINT
    END IF
  NEXT I%
  LOCATE 19, 31: PRINT "Standard  Deviations"
  LOCATE 21, 4: PRINT Header1$
  LOCATE 22, 4: PRINT Header2$
  LOCATE 23, 4: PRINT Header3$
  FOR I% = 1 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          LOCATE , 2
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT USING Format1$; I% - 1; J%;
          ELSE
            PRINT USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            IF I% = 1 THEN
              PRINT USING FormatA$; TPOverallStDevNormDefl(J%, K%) * 1000;
            ELSE
              PRINT USING FormatA$; OverallStDevNormDefl(I%, J%, K%) * 1000;
            END IF
          NEXT K%
          PRINT
        END IF
      NEXT J%
      PRINT
    END IF
  NEXT I%
```

```
  LOCATE 34, 29: PRINT "Coefficient of Variation"
  LOCATE 36, 4: PRINT Header1$
  LOCATE 37, 4: PRINT Header2$
  LOCATE 38, 4: PRINT Header3$
  FOR I% = 1 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          LOCATE , 2
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT USING Format1$; I% - 1; J%;
          ELSE
            PRINT USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            IF I% = 1 THEN
              PRINT USING "  ###.##%"; TPOverallCVNormDefl(J%, K%);
            ELSE
              PRINT USING "  ###.##%"; OverallCVNormDefl(I%, J%, K%);
            END IF
          NEXT K%
          PRINT
        END IF
      NEXT J%
      PRINT
    END IF
  NEXT I%
  DO
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 81       'PgDn
        WIDTH 80, 25
        EXIT DO
      CASE 60       'F2:PrtSc
        CALL PrtSc0(1)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB ShowSubSectionStats STATIC
  SCREEN 0: WIDTH 80, 50: WindowType% = 1
  Header1$ = "Test  Drop   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor"
  Header2$ = "Loc.   Ht       1        2        3        4        5        6        7"
  Header3$ = "----  ----   ------   ------   ------   ------   ------   ------   ------"
  Format1$ = "  #     # "
  Format2$ = "          # "
  IF English% THEN
    FormatA$ = "  ##.####"
  ELSE
    FormatA$ = "  #.#####"
  END IF
  SubSect% = 1
  DO
    CLS
    CALL ScreenBorder
    CALL TitleColor
    IF MatlCode%(1) = 700 THEN
      Title$ = "  CORRECTED Deflection Statistics - Subsection " + QPTrim$(STR$(SubSect%)) + "  "
    ELSE
      Title$ = "  Deflection Statistics - Subsection " + QPTrim$(STR$(SubSect%)) + "  "
    END IF
    TL% = LEN(Title$)
    Col% = ((80 - TL%) / 2) + 1
    LOCATE 2, Col%: PRINT Title$
    CALL HiliteColor
```

```
    LOCATE 4, 22: PRINT LEFT$(File$, 7)
    CALL NormalColor
    LOCATE 4, 5: PRINT "Data for section "
    IF English% THEN ZText$ = " (mils/kip)" ELSE ZText$ = " (microns/kPa)"
    LOCATE 4, 35: PRINT "Mean  Values"; ZText$
    StatusLine$ = "                                                PgDn"
    IF MatlCode%(1) = 700 THEN
      LOCATE 48, 10: PRINT "Note: Only sensor 1 deflections are corrected."
    END IF
    LOCATE 50, 4: PRINT StatusLine$;
    IF NumSubSect% > 1 THEN
      IF SubSect% < NumSubSect% THEN
        PRINT "  Ctrl-PgDn ";
      ELSE
        PRINT SPACE$(12);
      END IF
      IF SubSect% > 1 THEN
        PRINT " Ctrl-PgUp ";
      ELSE
        PRINT SPACE$(11);
      END IF
    END IF
    LOCATE 6, 4: PRINT Header1$
    LOCATE 7, 4: PRINT Header2$
    LOCATE 8, 4: PRINT Header3$
    FOR I% = 2 TO MaxLocations%
      IF LocationList%(I%) THEN
        PrintLocation% = False%
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            LOCATE , 2
            IF NOT PrintLocation% THEN
              PrintLocation% = True%
              PRINT USING Format1$; I% - 1; J%;
            ELSE
              PRINT USING Format2$; J%;
            END IF
            FOR K% = 1 TO NumDeflectors%
              PRINT USING FormatA$; SubSectionMeanDefl(I%, J%, K%, SubSect%) * 1000;
            NEXT K%
            PRINT
          END IF
        NEXT J%
        PRINT
      END IF
    NEXT I%
    LOCATE 19, 31: PRINT "Standard  Deviations"
    LOCATE 21, 4: PRINT Header1$
    LOCATE 22, 4: PRINT Header2$
    LOCATE 23, 4: PRINT Header3$
    FOR I% = 2 TO MaxLocations%
      IF LocationList%(I%) THEN
        PrintLocation% = False%
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            LOCATE , 2
            IF NOT PrintLocation% THEN
              PrintLocation% = True%
              PRINT USING Format1$; I% - 1; J%;
            ELSE
              PRINT USING Format2$; J%;
            END IF
            FOR K% = 1 TO NumDeflectors%
              PRINT USING FormatA$; SubSectionStDevDefl(I%, J%, K%, SubSect%) * 1000;
            NEXT K%
            PRINT
          END IF
        NEXT J%
        PRINT
      END IF
    NEXT I%
    LOCATE 34, 29: PRINT "Coefficient of Variation"
    LOCATE 36, 4: PRINT Header1$
```

```
    LOCATE 37, 4: PRINT Header2$
    LOCATE 38, 4: PRINT Header3$
    FOR I% = 2 TO MaxLocations%
      IF LocationList%(I%) THEN
        PrintLocation% = False%
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            LOCATE , 2
            IF NOT PrintLocation% THEN
              PrintLocation% = True%
              PRINT USING Format1$; I% - 1; J%;
            ELSE
              PRINT USING Format2$; J%;
            END IF
            FOR K% = 1 TO NumDeflectors%
              PRINT USING "  ###.###"; SubSectionCVDefl(I%, J%, K%, SubSect%);
            NEXT K%
            PRINT
          END IF
        NEXT J%
        PRINT
      END IF
    NEXT I%
    DO
      DO
        sdr$ = INKEY$
      LOOP WHILE sdr$ = ""
      KCode% = KeyCode%(sdr$)
      SELECT CASE KCode%
        CASE 81         'PgDn
          WIDTH 80, 25
          EXIT SUB
        CASE 60         'F2:PrtSc
          CALL PrtSc0(1)
          LPRINT CHR$(12);
        CASE 132            'Ctrl-PgUp
          IF SubSect% > 1 THEN
            SubSect% = SubSect% - 1
            EXIT DO
          ELSE
            REDIM PUText$(1)
            PUText$(1) = "This is the first available subsection.  Ctrl-PgUp is invalid..."
            CALL PopupError
          END IF
        CASE 118            'Ctrl-PgDn
          IF SubSect% < NumSubSect% THEN
            SubSect% = SubSect% + 1
            EXIT DO
          ELSE
            REDIM PUText$(1)
            PUText$(1) = "This is the last available subsection.  Ctrl-PgDn is invalid..."
            CALL PopupError
          END IF
        CASE ELSE
          'do nothing
      END SELECT
    LOOP
  LOOP
END SUB

SUB ShowTPFlexStats STATIC
  SCREEN 0: WIDTH 80, 25: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  FLEXIBLE Pavement Thickness Statistics - Test Pits  "
  LOCATE 2, 41 - LEN(Title$) / 2: PRINT Title$
  CALL HiliteColor
  LOCATE 4, 22: PRINT LEFT$(File$, 7)
  CALL NormalColor
  StatusLine$ = "                                              PgDn"
  LOCATE 25, 4: PRINT StatusLine$;
  LOCATE 4, 5: PRINT "Data for section"
  Header1$ = "                               Subgrade       Effective"
```

```
  Header2$ = " Height       Station        Modulus          SN"
  Header3$ = "--------     ---------     ----------     ----------"
  Format1$ = "   ##           ###          ######          ##.##"
  Format2$ = "                ###          ######          ##.##"
  Col% = 41 - LEN(Header1$) / 2
  LOCATE 7, Col%: PRINT Header1$
  LOCATE , Col%: PRINT Header2$
  LOCATE , Col%: PRINT Header3$
  FOR J% = 1 TO MaxHeight%
    IF HeightList%(J%) THEN
      FOR I% = 1 TO NumTPStations%
        IF I% = 1 THEN
          LOCATE , Col%: PRINT USING Format1$; J%; TPStation%(I%); TPBackCalcEsg(I%, J%); TPEquivalentSN(I%, J%)
        ELSE
          LOCATE , Col%: PRINT USING Format2$; TPStation%(I%); TPBackCalcEsg(I%, J%); TPEquivalentSN(I%, J%)
        END IF
      NEXT I%
      PRINT
    END IF
  NEXT J%
  DO
    DO
      sdr$ = INKEY$
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 81       'PgDn
        EXIT DO
      CASE 60       'F2:PrtSc
        CALL PrtSc0(1)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

SUB ShowTPRigidStats STATIC
  SCREEN 0: WIDTH 80, 25: WindowType% = 1: CLS
  CALL ScreenBorder
  CALL TitleColor
  Title$ = "  RIGID Pavement Thickness Statistics - Test Pits  "
  LOCATE 2, 41 - LEN(Title$) / 2: PRINT Title$
  CALL HiliteColor
  LOCATE 4, 22: PRINT LEFT$(File$, 7)
  CALL NormalColor
  StatusLine$ = "                                                PgDn"
  LOCATE 25, 4: PRINT StatusLine$;
  LOCATE 4, 5: PRINT "Data for section"
  Header1$ = "                                 Volumetric       Effective"
  Header2$ = " Height       Station              k             Thickness"
  Header3$ = "--------     ---------     ------------     ----------"
  Format1$ = "   ##           ###             ####             ##.##"
  Format2$ = "                ###             ####             ##.##"
  Col% = 41 - LEN(Header1$) / 2
  LOCATE 7, Col%: PRINT Header1$
  LOCATE , Col%: PRINT Header2$
  LOCATE , Col%: PRINT Header3$
  FOR J% = 1 TO MaxHeight%
    IF HeightList%(J%) THEN
      FOR I% = 1 TO NumTPStations%
        IF I% = 1 THEN
          LOCATE , Col%: PRINT USING Format1$; J%; TPStation%(I%); TPVolumetricK(I%, J%); TPRigidThickness(I%, J%)
        ELSE
          LOCATE , Col%: PRINT USING Format2$; TPStation%(I%); TPVolumetricK(I%, J%); TPRigidThickness(I%, J%)
        END IF
      NEXT I%
      PRINT
    END IF
  NEXT J%
  DO
    DO
      sdr$ = INKEY$
```

```
    LOOP WHILE sdr$ = ""
    KCode% = KeyCode%(sdr$)
    SELECT CASE KCode%
      CASE 81       'PgDn
        EXIT DO
      CASE 60       'F2:PrtSc
        CALL PrtSc0(1)
        LPRINT CHR$(12);
      CASE ELSE
        'do nothing
    END SELECT
  LOOP
END SUB

FUNCTION TPNormRigidDefl (hx, I%, J%) STATIC
' Westergaard solution:
  Gamma# = .57721566649#  'Euler's constant
  L = ((1000 * Modulus(1) * hx ^ 3) / (12 * (1 - .15 ^ 2) * TPVolumetricK(I%, J%))) ^ (1 / 4)
  C1 = 1 + (1 / (2 * Pi)) * (LOG(TLPR / (2 * L)) + Gamma# - 1.25) * (TLPR / L) ^ 2
  TPNormRigidDefl = 1 / (8 * TPVolumetricK(I%, J%) * L ^ 2) * C1
END FUNCTION

SUB WriteOutlierMsgs STATIC
  ThresholdDeviation = 2
  LastStation% = 0
  Header1$ = "                                          Number of"
  Header2$ = " Station       Height       Sensor       Std. Dev."
  Header3$ = "---------     --------     --------     -----------"
  Format1$ = "  ####           ##           ##          ###.##"
  Format2$ = "  #### (TP)      ##           ##          ###.##"
  GOSUB PrintTitle
  TP1Outliers% = 0
  TP2Outliers% = 0
  REDIM NumOutliers%(NumSubSect%)
  SectionOutliers% = False%
  FOR L% = 1 TO NumSubSect%
    FirstStation% = LastStation% + 1
    LastStation% = StationCount%(L%) + FirstStation% - 1
    SSTPOutliers% = False%
    SubSectionOutliers% = False%
    IF NumSubSect% > 1 THEN
      PRINT #2, SPC(10); USING "Subsection #"; L%
      Counter% = Counter% + 1
    END IF
    GOSUB PrintHeader
    TC% = Counter%
    IF L% = 1 THEN
      FOR I% = 1 TO NumTPStations%
        IF TPStation%(I%) < 0 THEN
          FOR J% = 1 TO MaxHeight%
            IF HeightList%(J%) THEN
              FOR K% = 1 TO NumDeflectors%
                IF ABS(TPDeviation(I%, J%, K%)) > ThresholdDeviation THEN
                  TP1Outliers% = TP1Outliers% + 1
                  SSTPOutliers% = True%
                  Counter% = Counter% + 1
                  PRINT #2, SPC(10); USING Format2$; TPStation%(I%); J%; K%; TPDeviation(I%, J%, K%)
                END IF
              NEXT K%
            END IF
          NEXT J%
        END IF
      NEXT I%
    END IF
    IF Counter% <> TC% THEN
      PRINT #2, ""
      Counter% = Counter% + 1
    END IF
    FOR I% = FirstStation% TO LastStation%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          FOR K% = 1 TO NumDeflectors%
            IF ABS(Deviation(I%, J%, K%)) > ThresholdDeviation THEN
```

```
            NumOutliers%(L%) = NumOutliers%(L%) + 1
            SubSectionOutliers% = True%
            SectionOutliers% = True%
            Counter% = Counter% + 1
            PRINT #2, SPC(10); USING Format1$; Station%(I%); J%; K%; Deviation(I%, J%, K%)
          END IF
          IF Counter% > 55 THEN
            PRINT #2, CHR$(12);
            GOSUB PrintTitle
            GOSUB PrintHeader
          END IF
        NEXT K%
      END IF
    NEXT J%
  NEXT I%
  IF Counter% <> TC% THEN
    PRINT #2, ""
    Counter% = Counter% + 1
  END IF
  IF L% = NumSubSect% THEN
    FOR I% = 1 TO NumTPStations%
      IF TPStation%(I%) > 500 THEN
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            FOR K% = 1 TO NumDeflectors%
              IF ABS(TPDeviation(I%, J%, K%)) > ThresholdDeviation THEN
                TP2Outliers% = TP2Outliers% + 1
                SSTPOutliers% = True%
                Counter% = Counter% + 1
                PRINT #2, SPC(10); USING Format2$; TPStation%(I%); J%; K%; TPDeviation(I%, J%, K%)
              END IF
              IF Counter% > 55 THEN
                PRINT #2, CHR$(12);
                GOSUB PrintTitle
                GOSUB PrintHeader
              END IF
            NEXT K%
          END IF
        NEXT J%
      END IF
    NEXT I%
  END IF
  IF NOT SSTPOutliers% AND NOT SubSectionOutliers% THEN
    PRINT #2, ""
    PRINT #2, SPC(14); "No deflection data for this subsection is more than"
    PRINT #2, SPC(14); USING "#.# standard deviations from the subsection mean."; ThresholdDeviation
    PRINT #2, ""
    Counter% = Counter% + 4
  END IF
  IF Counter% > 42 AND L% <> NumSubSect% THEN
    PRINT #2, CHR$(12);
    GOSUB PrintTitle
  END IF
NEXT L%
IF Counter% <> 0 THEN PRINT #2, CHR$(12);
EXIT SUB

PrintHeader:
  PRINT #2, ""
  PRINT #2, SPC(10); Header1$
  PRINT #2, SPC(10); Header2$
  PRINT #2, SPC(10); Header3$
  Counter% = Counter% + 4
  RETURN

PrintTitle:
  PRINT #2, ""
  PRINT #2, SPC(26); "Outlier Statistics - "; LEFT$(File$, 7)
  PRINT #2, ""
  Counter% = 3
  RETURN

END SUB
```

```
SUB WriteOverallStatistics STATIC
  Title$ = "UNCORRECTED Overall Deflection Statistics"
  Header1$ = "Test  Drop    Sensor    Sensor    Sensor    Sensor    Sensor    Sensor    Sensor"
  Header2$ = "Loc.   Ht       1         2         3         4         5         6         7"
  Header3$ = "----  ----   ------    ------    ------    ------    ------    ------    ------"
  Format1$ = "    #     # "
  Format2$ = "          # "
  IF English% THEN
    FormatA$ = "  ##.####"
  ELSE
    FormatA$ = "  #.#####"
  END IF
  Col% = 41 - LEN(Title$) / 2
  PRINT #2, ""
  PRINT #2, SPC(22); "Summary of Data for section "; LEFT$(File$, 7)
  PRINT #2, SPC(22); "Analyzed by: "; QPTrim$(Analyst$); " on "; DATE$
  PRINT #2, ""
  PRINT #2, SPC(Col% - 1); Title$
  PRINT #2, ""
  IF English% THEN ZText$ = " (mils/kip)" ELSE ZText$ = " (microns/kPa)"
  PRINT #2, SPC(35); "Mean Values"; ZText$
  PRINT #2, ""
  PRINT #2, SPC(3); Header1$
  PRINT #2, SPC(3); Header2$
  PRINT #2, SPC(3); Header3$
  FOR I% = 1 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT #2, USING Format1$; I% - 1; J%;
          ELSE
            PRINT #2, USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            IF I% = 1 THEN
              PRINT #2, USING FormatA$; TPOverallMeanNormDefl(J%, K%) * 1000;
            ELSE
              PRINT #2, USING FormatA$; OverallMeanNormDefl(I%, J%, K%) * 1000;
            END IF
          NEXT K%
          PRINT #2, ""
        END IF
      NEXT J%
      PRINT #2, ""
    END IF
  NEXT I%
  PRINT #2, ""
  PRINT #2, SPC(30); "Standard  Deviations"
  PRINT #2, ""
  PRINT #2, SPC(3); Header1$
  PRINT #2, SPC(3); Header2$
  PRINT #2, SPC(3); Header3$
  FOR I% = 1 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT #2, USING Format1$; I% - 1; J%;
          ELSE
            PRINT #2, USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            IF I% = 1 THEN
              PRINT #2, USING FormatA$; TPOverallStDevNormDefl(J%, K%) * 1000;
            ELSE
              PRINT #2, USING FormatA$; OverallStDevNormDefl(I%, J%, K%) * 1000;
            END IF
          NEXT K%
```

```
        PRINT #2, ""
      END IF
    NEXT J%
    PRINT #2, ""
  END IF
NEXT I%
PRINT #2, ""
PRINT #2, SPC(28); "Coefficient of Variation"
PRINT #2, ""
PRINT #2, SPC(3); Header1$
PRINT #2, SPC(3); Header2$
PRINT #2, SPC(3); Header3$
FOR I% = 1 TO MaxLocations%
  IF LocationList%(I%) THEN
    PrintLocation% = False%
    FOR J% = 1 TO MaxHeight%
      IF HeightList%(J%) THEN
        IF NOT PrintLocation% THEN
          PrintLocation% = True%
          PRINT #2, USING Format1$; I% - 1; J%;
        ELSE
          PRINT #2, USING Format2$; J%;
        END IF
        FOR K% = 1 TO NumDeflectors%
          IF I% = 1 THEN
            PRINT #2, USING "  ###.##%"; TPOverallCVNormDefl(J%, K%);
          ELSE
            PRINT #2, USING "  ###.##%"; OverallCVNormDefl(I%, J%, K%);
          END IF
        NEXT K%
        PRINT #2, ""
      END IF
    NEXT J%
    PRINT #2, ""
  END IF
NEXT I%
PRINT #2, CHR$(12);
END SUB

SUB WriteRunningComments STATIC
  AnyComments% = False%
  IF QPTrim$(Comment$) <> "" THEN       '***** improve this *****
    AnyComments% = True%
    TmpComment$ = QPTrim$(Comment$)
    NumChars% = LEN(TmpComment$)
    FOR I% = NumChars% TO 1 STEP -1
      IF MID$(TmpComment$, I%, 1) <> CHR$(13) THEN
        NumChars2% = I% + 1
        EXIT FOR
      END IF
    NEXT I%
    IF NumChars% < NumChars2% THEN NumChars2% = NumChars%
    FOR I% = 1 TO NumChars2%            'add LF to CR characters
      PRINT #2, MID$(TmpComment$, I%, 1);
      IF MID$(TmpComment$, I%, 1) = CHR$(13) THEN PRINT #2, CHR$(10);
    NEXT I%
  END IF
  IF AnyComments% THEN PRINT #2, "": PRINT #2, ""
  IF SummaryOfResults% THEN
    NumHeights% = 0
    FOR I% = 1 TO 4
      IF HeightList%(I%) THEN NumHeights% = NumHeights% + 1
    NEXT I%
    HSCombo% = NumHeights% * (NumStations% + NumTPStations%)
    HSSCombo% = HSCombo% * NumDeflectors%
    PRINT #2, SPC(31); "Summary of Results"
    PRINT #2, ""
    PRINT #2, "  Section uniformity:"
    IF NumSubSect% = 1 THEN
      PRINT #2, SPC(4); "NO Subsections were identified within the section."
    ELSE
      PRINT #2, SPC(4); "Subsections were identified within the section."
      FOR I% = 1 TO NumSubSect%
```

```
        IF I% = 1 THEN
          PRINT #2, USING "      Subsection 1 boundaries occur at   0 ft. and ### ft."; SubSectEnd%(I%)
        ELSE
          PRINT #2, USING "        Subsection # boundaries occur at ### ft. and ### ft."; I%; SubSectEnd%(I% - 1);
SubSectEnd%(I%)
        END IF
      NEXT I%
      PRINT #2, ""
      PRINT #2, SPC(4); "Comparing subsections:"
      FOR I% = 1 TO NumSubSect% - 1
        PRINT #2, USING "      Subsections # and #: "; I%; I% + 1;
        IF NOT Status%(I%, 1) THEN PRINT #2, "UN";
        PRINT #2, "EQUAL means and ";
        IF NOT Status%(I%, 2) THEN PRINT #2, "UN";
        PRINT #2, "EQUAL variances."
      NEXT I%
    END IF
    PRINT #2, ""
    PRINT #2, USING "  Outliers - Test pits: ## combinations at each test pit"; NumHeights% * NumDeflectors%
    IF NumTPStations% = 0 THEN
      PRINT #2, SPC(4); "NO Test pit data was present."
    ELSE
      IF TP1Outliers% THEN
        PRINT #2, USING "    ## height/sensor combinations at TP 1 DO NOT appear"; TP1Outliers%
        PRINT #2, SPC(8); "representative of section data."
      ELSE
        PRINT #2, SPC(4); "All TP 1 data appears representative of section data."
      END IF
      IF TP2Outliers% THEN
        PRINT #2, USING "    ## height/sensor combinations at TP 2 DO NOT appear"; TP2Outliers%
        PRINT #2, SPC(8); "representative of section data."
      ELSE
        PRINT #2, SPC(4); "All TP 2 data appears representative of section data."
      END IF
    END IF
    PRINT #2, ""
    PRINT #2, USING "  Outliers - Section data: ### total combinations within the section"; HSSCombo%
    IF SectionOutliers% THEN
      FOR I% = 1 TO NumSubSect%
        IF NumOutliers%(I%) > 0 THEN
          PRINT #2, USING "    ### height/sensor/station combinations are data outliers in subsection #."; NumOutliers%(I%);
I%
        ELSE
          PRINT #2, USING "    There are NO data outliers within subsection #."; I%
        END IF
      NEXT I%
    ELSE
      PRINT #2, SPC(4); "There are NO data outliers within the section data."
    END IF
    IF SummaryOfResults2% THEN
      PRINT #2, ""
      PRINT #2, USING "  Structural capacity - Test pits: # combinations at each test pit"; NumHeights%
      IF TP1Capacity% THEN
        PRINT #2, USING "    # height(s) for TP 1 are NOT within the range of expected values."; TP1Capacity%
      ELSE
        PRINT #2, SPC(4); "All results for TP 1 are within the range of expected values."
      END IF
      IF TP2Capacity% THEN
        PRINT #2, USING "    # height(s) for TP 2 are NOT within the range of expected values."; TP2Capacity%
      ELSE
        PRINT #2, SPC(4); "All results for TP 2 are within the range of expected values."
      END IF
      PRINT #2, ""
      PRINT #2, USING "  Structural capacity - Section data: ### total combinations within the section"; HSCombo%
      IF SectionCapacity% THEN
        PRINT #2, USING "    ## height/station combinations are NOT within the range of expected values."; SectionCapacity%
      ELSE
        PRINT #2, SPC(4); "All results are within the range of expected values."
      END IF
      IF MatlCode%(1) = 700 THEN
        PRINT #2, ""
        PRINT #2, "  Subgrade response:"
        IF Linearity%(1) > 0 THEN PRINT #2, USING "      ### height/station combinations exhibit linear response.";
```

```
Linearity%(1)
        IF Linearity%(2) > 0 THEN PRINT #2, USING "     ### height/station combinations exhibit slightly non-linear
response."; Linearity%(2)
        IF Linearity%(3) > 0 THEN PRINT #2, USING "     ### height/station combinations exhibit distinctly non-linear
response."; Linearity%(3)
      END IF
    END IF
  END IF
  IF AnyComments% OR SummaryOfResults% THEN PRINT #2, CHR$(12);
END SUB

SUB WriteStructuralStats STATIC
  TP1Capacity% = 0
  TP2Capacity% = 0
  SectionCapacity% = 0
  PRINT #2, ""
  PRINT #2, SPC(18); "Pavement Construction Information - "; LEFT$(File$, 7)
  Header1$ = " Material                                    Layer"
  Header2$ = "   Code            Material Name            Thickness"
  Header3$ = "----------     ------------------------     ----------"
  Format1$ = "   ####        \                       \        ##.#"
  Counter% = 2
  GOSUB PrintHeader2
  FOR I% = 1 TO NumLayers%
    PRINT #2, SPC(10); USING Format1$; MatlCode%(I%); RIGHT$(MaterialList$(SelCode%(I%)), 24); Thickness(I%)
  NEXT I%
  PRINT #2, ""
  Counter% = Counter% + NumLayers% + 1
  IF MatlCode%(1) = 700 THEN
    PRINT #2, USING "          Depth to rigid foundation: ###.# ft."; DepthRigidLayer / 12
    PRINT #2, ""
    Counter% = Counter% + 2
  END IF
  IF SummaryOfResults2% THEN
    Counter% = Counter% + NumLayers% + 1
    IF MatlCode%(1) = 700 THEN
      Title$ = SPACE$(19) + "FLEXIBLE Pavement Thickness Data - " + LEFT$(File$, 7)
      PRINT #2, ""
      PRINT #2, Title$
      PRINT #2, SPC(13); "(comparison of each calculation to the expected value)"
      PRINT #2, ""
      PRINT #2, SPC(10); USING "Minimum expected SN value: ##.##"; LoSN
      PRINT #2, SPC(10); USING "Maximum expected SN value: ##.##"; HiSN
      Counter% = Counter% + 6
      Header1$ = "                                Effective"
      Header2$ = " Height       Station            SN"
      Header3$ = "--------     ---------     -----------"
      Format1$ = "   ##          ####           ##.##"
      Format2$ = "   ##          #### (TP)      ##.##"
      GOSUB PrintHeader2
      TC% = Counter%
      FOR I% = 1 TO NumTPStations%
        IF TPStation%(I%) < 0 THEN
          FOR J% = 1 TO MaxHeight%
            IF HeightList%(J%) THEN
              IF TPEquivalentSN(I%, J%) < LoSN OR TPEquivalentSN(I%, J%) > HiSN THEN
                TP1Capacity% = TP1Capacity% + 1
                Counter% = Counter% + 1
                PRINT #2, SPC(10); USING Format2$; J%; TPStation%(I%); TPEquivalentSN(I%, J%)
              END IF
            END IF
          NEXT J%
        END IF
      NEXT I%
      IF Counter% <> TC% THEN
        PRINT #2, ""
        Counter% = Counter% + 1
      END IF
      FOR I% = 1 TO NumStations%
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            IF EquivalentSN(I%, J%) < LoSN OR EquivalentSN(I%, J%) > HiSN THEN
              SectionCapacity% = SectionCapacity% + 1
```

```
          Counter% = Counter% + 1
          PRINT #2, SPC(10); USING Format1$; J%; Station%(I%); EquivalentSN(I%, J%)
        END IF
      END IF
      IF Counter% > 55 THEN GOSUB NewPage2
    NEXT J%
  NEXT I%
  IF Counter% <> TC% THEN
    PRINT #2, ""
    Counter% = Counter% + 1
  END IF
  FOR I% = 1 TO NumTPStations%
    IF TPStation%(I%) > 500 THEN
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          IF TPEquivalentSN(I%, J%) < LoSN OR TPEquivalentSN(I%, J%) > HiSN THEN
            TP2Capacity% = TP2Capacity% + 1
            Counter% = Counter% + 1
            PRINT #2, SPC(10); USING Format2$; J%; TPStation%(I%); TPEquivalentSN(I%, J%)
          END IF
        END IF
        IF Counter% > 55 THEN GOSUB NewPage2
      NEXT J%
    END IF
  NEXT I%
  IF Counter% = TC% THEN
    PRINT #2, ""
    PRINT #2, SPC(11); "No predicted SN values fall outside the expected range..."
  END IF
  PRINT #2, CHR$(12);
  Header1$ = "                                  Subgrade      Effective "
  Header2$ = " Subsection      Station      Modulus          SN      "
  Header3$ = STRING$(54, "-")
  Format1$ = "     ##           ####        ######          ##.##"
  Format2$ = "                  ####        ######          ##.##"
  Format3$ = "    (TP)          ####        ######          ##.##"
  Footer1$ = "    Subsection # Overall Mean:######          ##.##"
  Footer2$ = "           Standard Deviation:######          ##.##"
  Footer3$ = "           Coeff Of Variation:####.##%       ###.##%"
  FOR J% = 1 TO MaxHeight%
    LastStation% = 0
    IF HeightList%(J%) THEN
      PRINT #2, ""
      PRINT #2, SPC(16); "FLEXIBLE Pavement Thickness Statistics - "; LEFT$(File$, 7)
      PRINT #2, ""
      PRINT #2, USING "     Drop height #"; J%
      GOSUB PrintHeader2
      IF NumTPStations% <= 0 THEN
        PRINT #2, SPC(10); "No test pit data found, therefore no results exist..."
      ELSE
        PRINT #2, SPC(10); USING Format3$; TPStation%(1); TPBackCalcEsg(1, J%); TPEquivalentSN(1, J%)
      END IF
      FOR I% = 1 TO NumSubSect%
        FirstStation% = LastStation% + 1
        LastStation% = StationCount%(I%) + FirstStation% - 1
        PRINT #2, SPC(10); Header3$
        FOR L% = FirstStation% TO LastStation%
          IF L% = FirstStation% THEN
            PRINT #2, SPC(10); USING Format1$; I%; Station%(L%); BackCalcEsg(L%, J%); EquivalentSN(L%, J%)
          ELSE
            PRINT #2, SPC(10); USING Format2$; Station%(L%); BackCalcEsg(L%, J%); EquivalentSN(L%, J%)
          END IF
        NEXT L%
      NEXT I%
      IF NumTPStations% >= 2 THEN
        PRINT #2, SPC(10); Header3$
        PRINT #2, SPC(10); USING Format3$; TPStation%(2); TPBackCalcEsg(2, J%); TPEquivalentSN(2, J%)
      END IF
      FOR I% = 1 TO NumSubSect%
        PRINT #2, SPC(10); Header3$
        PRINT #2, SPC(10); USING Footer1$; I%; BCEsgMean(I%, J%); EqSNMean(I%, J%)
        PRINT #2, SPC(10); USING Footer2$; BCEsgStDev(I%, J%); EqSNStDev(I%, J%)
        PRINT #2, SPC(10); USING Footer3$; BCEsgCV(I%, J%); EqSNCV(I%, J%)
```

```
        NEXT I%
        PRINT #2, SPC(10); Header3$
        PRINT #2, CHR$(12);
      END IF
    NEXT J%
  ELSEIF MatlCode%(1) = 730 THEN
    Title$ = SPACE$(21) + "RIGID Pavement Thickness Data - " + LEFT$(File$, 7)
    PRINT #2, ""
    PRINT #2, Title$
    PRINT #2, SPC(13); "(comparison of each calculation to the expected value)"
    PRINT #2, ""
    PRINT #2, SPC(10); USING "Minimum expected thickness: ##.##"; LoThick
    PRINT #2, SPC(10); USING "Maximum expected thickness: ##.##"; HiThick
    Counter% = Counter% + 6
    Header1$ = "                              Effective"
    Header2$ = " Height       Station         Thickness"
    Header3$ = "--------     ---------     -----------"
    Format1$ = "   ##          ####           ##.##"
    Format2$ = "   ##          #### (TP)      ##.##"
    GOSUB PrintHeader2
    TC% = Counter%
    FOR I% = 1 TO NumTPStations%
      IF TPStation%(I%) < 0 THEN
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            IF TPRigidThickness(I%, J%) < LoThick OR TPRigidThickness(I%, J%) > HiThick THEN
              TP1Capacity% = TP1Capacity% + 1
              Counter% = Counter% + 1
              PRINT #2, SPC(10); USING Format2$; J%; TPStation%(I%); TPRigidThickness(I%, J%)
            END IF
          END IF
        NEXT J%
      END IF
    NEXT I%
    IF Counter% <> TC% THEN
      PRINT #2, ""
      Counter% = Counter% + 1
    END IF
    FOR I% = 1 TO NumStations%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          IF RigidThickness(I%, J%) < LoThick OR RigidThickness(I%, J%) > HiThick THEN
            SectionCapacity% = SectionCapacity% + 1
            Counter% = Counter% + 1
            PRINT #2, SPC(10); USING Format1$; J%; Station%(I%); RigidThickness(I%, J%)
          END IF
        END IF
        IF Counter% > 55 THEN GOSUB NewPage2
      NEXT J%
    NEXT I%
    IF Counter% <> TC% THEN
      PRINT #2, ""
      Counter% = Counter% + 1
    END IF
    FOR I% = 1 TO NumTPStations%
      IF TPStation%(I%) > 500 THEN
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            IF TPRigidThickness(I%, J%) < LoThick OR TPRigidThickness(I%, J%) > HiThick THEN
              TP2Capacity% = TP2Capacity% + 1
              Counter% = Counter% + 1
              PRINT #2, SPC(10); USING Format2$; J%; TPStation%(I%); TPRigidThickness(I%, J%)
            END IF
          END IF
          IF Counter% > 55 THEN GOSUB NewPage2
        NEXT J%
      END IF
    NEXT I%
    IF Counter% = TC% THEN
      PRINT #2, ""
      PRINT #2, SPC(8); "No predicted thickness values fall outside the expected range..."
    END IF
    PRINT #2, CHR$(12);
```

```
      Header1$ = "                                Volumetric      Effective "
      Header2$ = " Subsection      Station          k            Thickness "
      Header3$ = STRING$(56, "-")
      Format1$ = "     ##          ####           ####           ##.##"
      Format2$ = "                 ####           ####           ##.##"
      Format3$ = "    (TP)         ####           ####           ##.##"
      Footer1$ = "      Subsection # Overall Mean: ####           ##.##"
      Footer2$ = "             Standard Deviation: ####           ##.##"
      Footer3$ = "             Coeff Of Variation: ###.##%       ###.##%"
      FOR J% = 1 TO MaxHeight%
        LastStation% = 0
        IF HeightList%(J%) THEN
          PRINT #2, ""
          PRINT #2, SPC(17); "RIGID Pavement Thickness Statistics - "; LEFT$(File$, 7)
          PRINT #2, ""
          PRINT #2, USING "     Drop height #"; J%
          GOSUB PrintHeader2
          IF NumTPStations% <= 0 THEN
            PRINT #2, SPC(10); "No test pit data found, therefore no results exist..."
          ELSE
            PRINT #2, SPC(10); USING Format3$; TPStation%(1); TPVolumetricK(1, J%); TPRigidThickness(1, J%)
          END IF
          FOR I% = 1 TO NumSubSect%
            FirstStation% = LastStation% + 1
            LastStation% = StationCount%(I%) + FirstStation% - 1
            PRINT #2, SPC(10); Header3$
            FOR L% = FirstStation% TO LastStation%
              IF L% = FirstStation% THEN
                PRINT #2, SPC(10); USING Format1$; I%; Station%(L%); VolumetricK(L%, J%); RigidThickness(L%, J%)
              ELSE
                PRINT #2, SPC(10); USING Format2$; Station%(L%); VolumetricK(L%, J%); RigidThickness(L%, J%)
              END IF
            NEXT L%
          NEXT I%
          IF NumTPStations% >= 2 THEN
            PRINT #2, SPC(10); Header3$
            PRINT #2, SPC(10); USING Format3$; TPStation%(2); TPVolumetricK(2, J%); TPRigidThickness(2, J%)
          END IF
          FOR I% = 1 TO NumSubSect%
            PRINT #2, SPC(10); Header3$
            PRINT #2, SPC(10); USING Footer1$; I%; VolKMean(I%, J%); RigThickMean(I%, J%)
            PRINT #2, SPC(10); USING Footer2$; VolKStDev(I%, J%); RigThickStDev(I%, J%)
            PRINT #2, SPC(10); USING Footer3$; VolKCV(I%, J%); RigThickCV(I%, J%)
          NEXT I%
          PRINT #2, SPC(10); Header3$
          PRINT #2, CHR$(12);
        END IF
      NEXT J%
    END IF
  ELSE              'analysis FAILED!!!!!
    PRINT #2, ""
    PRINT #2, SPC(10); "Structural Analysis has FAILED!"
    PRINT #2, CHR$(12);
  END IF
  EXIT SUB

PrintHeader2:
  PRINT #2, ""
  PRINT #2, SPC(10); Header1$
  PRINT #2, SPC(10); Header2$
  PRINT #2, SPC(10); Header3$
  Counter% = Counter% + 4
  RETURN

NewPage2:
  PRINT #2, CHR$(12);
  PRINT #2, ""
  PRINT #2, Title$
  PRINT #2, ""
  Counter% = 3
  GOSUB PrintHeader2
  TC% = Counter%
  RETURN
```

```
END SUB

SUB WriteSubsectionStats STATIC
  Header1$ = "Test  Drop   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor   Sensor"
  Header2$ = "Loc.   Ht       1        2        3        4        5        6        7"
  Header3$ = "----  ----   ------   ------   ------   ------   ------   ------   ------"
  Format1$ = "    #     # ##
  Format2$ = "                # "
  IF English% THEN
    FormatA$ = "  ##.####"
  ELSE
    FormatA$ = "  #.#####"
  END IF
  IF English% THEN ZText$ = " (mils/kip)" ELSE ZText$ = " (microns/kPa)"
  FOR SubSect% = 1 TO NumSubSect%
    SELECT CASE MatlCode%(1)
      CASE 700
        Title$ = "Flexible Pavement Deflection Statistics - " + LEFT$(File$, 7)
      CASE 730
        Title$ = "Rigid Pavement Deflection Statistics - " + LEFT$(File$, 7)
    END SELECT
    TL% = LEN(Title$)
    Col% = ((80 - TL%) / 2) + 1
    PRINT #2, ""
    PRINT #2, SPC(Col% - 1); Title$
    IF NumSubSect% > 1 THEN
      PRINT #2, SPC(32); "Subsection "; QPTrim$(STR$(SubSect%))
    END IF
    IF NumSubSect% > 1 THEN
      IF SubSect% = 1 THEN
        PRINT #2, SPC(22); "Subsection begins at station   0"
      ELSE
        PRINT #2, SPC(22); "Subsection begins at station"; SubSectEnd%(SubSect% - 1)
      END IF
      PRINT #2, SPC(22); "  Subsection ends at station"; SubSectEnd%(SubSect%)
    END IF
    PRINT #2, ""
    PRINT #2, SPC(35); "Mean  Values"; ZText$
    PRINT #2, ""
    IF MatlCode%(1) = 700 THEN PRINT #2, SPC(15); "CORRECTED"
    PRINT #2, SPC(3); Header1$
    PRINT #2, SPC(3); Header2$
    PRINT #2, SPC(3); Header3$
    FOR I% = 2 TO MaxLocations%
      IF LocationList%(I%) THEN
        PrintLocation% = False%
        FOR J% = 1 TO MaxHeight%
          IF HeightList%(J%) THEN
            IF NOT PrintLocation% THEN
              PrintLocation% = True%
              PRINT #2, USING Format1$; I% - 1; J%;
            ELSE
              PRINT #2, USING Format2$; J%;
            END IF
            FOR K% = 1 TO NumDeflectors%
              PRINT #2, USING FormatA$; SubSectionMeanDefl(I%, J%, K%, SubSect%) * 1000;
            NEXT K%
            PRINT #2, ""
          END IF
        NEXT J%
        PRINT #2, ""
      END IF
    NEXT I%
    PRINT #2, ""
    PRINT #2, SPC(30); "Standard  Deviations"
    PRINT #2, ""
    PRINT #2, SPC(3); Header1$
    PRINT #2, SPC(3); Header2$
    PRINT #2, SPC(3); Header3$
    FOR I% = 2 TO MaxLocations%
      IF LocationList%(I%) THEN
        PrintLocation% = False%
        FOR J% = 1 TO MaxHeight%
```

```
        IF HeightList%(J%) THEN
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT #2, USING Format1$; I% - 1; J%;
          ELSE
            PRINT #2, USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            PRINT #2, USING Format4$; SubSectionStDevDefl(I%, J%, K%, SubSect%) * 1000;
          NEXT K%
          PRINT #2, ""
        END IF
      NEXT J%
      PRINT #2, ""
    END IF
  NEXT I%
  PRINT #2, ""
  PRINT #2, SPC(28); "Coefficient of Variation"
  PRINT #2, ""
  PRINT #2, SPC(3); Header1$
  PRINT #2, SPC(3); Header2$
  PRINT #2, SPC(3); Header3$
  FOR I% = 2 TO MaxLocations%
    IF LocationList%(I%) THEN
      PrintLocation% = False%
      FOR J% = 1 TO MaxHeight%
        IF HeightList%(J%) THEN
          IF NOT PrintLocation% THEN
            PrintLocation% = True%
            PRINT #2, USING Format1$; I% - 1; J%;
          ELSE
            PRINT #2, USING Format2$; J%;
          END IF
          FOR K% = 1 TO NumDeflectors%
            PRINT #2, USING "  ###.##%"; SubSectionCVDefl(I%, J%, K%, SubSect%);
          NEXT K%
          PRINT #2, ""
        END IF
      NEXT J%
      PRINT #2, ""
    END IF
  NEXT I%
  PRINT #2, CHR$(12);
NEXT SubSect%
END SUB
```

```
DECLARE SUB LinearLeastSquaresFit (Num%, XI(), YI(), Constant!, Slope!, Rsqd!)
DECLARE SUB CHEVRON (WGT!, GNum%, ZCOMP!, NOpt%, CDEV!)
DECLARE SUB CALCIN (GNum%, CDEV!, NOpt%, L%)
DECLARE SUB COFE (LC%, P!)
DECLARE SUB BESSEL (NI%, XI!, Y!)
DECLARE SUB Part (Ct%)
DECLARE FUNCTION Log10 (Value!)                    'do not erase
'$INCLUDE: 'declare.inc'
'$INCLUDE: 'fdeclar1.inc'
'$INCLUDE: 'fdeclar2.inc'
'$INCLUDE: 'cmnblank.inc'
'$INCLUDE: 'fwdcheck.inc'
'$INCLUDE: 'chev.inc'

CONST True% = -1, False% = 0, MaxNumStations% = 50, MaxLocations% = 6
CONST MaxDeflectors% = 7, MaxNumPeaks% = 16, MaxHeights% = 4, MaxLayers% = 10
CONST MinPointsInSubSection% = 4, ThresholdAlpha = 95, MaxTimes% = 10
CONST Pi = 3.141593

SUB BESSEL (NI%, XI, Y) STATIC
  REDIM DD(20)
  IF XI <= 7 THEN
    X2 = XI / 2!
    FAC = -X2 * X2
    IF NI% <= 0 THEN
      C = 1!
      Y = C
      FOR I% = 1 TO 34
        C = FAC * C / CSNG(I% * I%)
        IF ABS(C) <= 1E-08 THEN EXIT SUB
        Y = Y + C
      NEXT I%
    END IF
    C = X2
    Y = C
    FOR I% = 1 TO 34
      C = FAC * C / CSNG(I% * (I% + 1))
      IF ABS(C) <= 1E-08 THEN EXIT SUB
      Y = Y + C
    NEXT I%
  ELSE
    IF NI% <= 0 THEN
      FOR I% = 1 TO 6
        DD(I%) = PZ(I%)
        DD(I% + 10) = QZ(I%)
      NEXT I%
    ELSE
      FOR I% = 1 TO 6
        DD(I%) = PO(I%)
        DD(I% + 10) = QO(I%)
      NEXT I%
    END IF
    T1 = 25! / XI
    T2 = T1 * T1
    P = DD(6) * T2 + DD(5)
    FOR I% = 1 TO 4
      J% = 5 - I%
      P = P * T2 + DD(J%)
    NEXT I%
    Q = DD(16) * T2 + DD(15)
    FOR I% = 1 TO 4
      J% = 5 - I%
      Q = Q * T2 + DD(J% + 10)
    NEXT I%
    Q = Q * T1
    T4 = SQR(XI * Pi)
    T6 = SIN(XI)
    T7 = COS(XI)
    IF NI% <= 0 THEN
      Y = ((P - Q) * T6 + (P + Q) * T7) / T4
    ELSE
      Y = ((P + Q) * T6 - (P - Q) * T7) / T4
    END IF
```

```
  END IF
END SUB

SUB CALCIN (GNum%, CDEV, NOpt%, L%) STATIC
  REDIM W(4)
  W(1) = .3478549
  W(2) = .6521452
  W(3) = .6521452
  W(4) = .3478549
  VL = 2! * V(L%)
  EL = (1! + V(L%)) / E(L%)
  VL1 = 1! - VL
  CSZ = 0!
  CST = 0!
  CSR = 0!
  CTR = 0!
  COMM = 0!
  CMU = 0!
  NTS1% = NTest%(GNum%) + 1
  ITS% = 1
  ARP = TLPR * PSI
  FOR I% = 1 TO 46
    RSZ = 0!                'INITIALIZE THE SUB-INTEGRALS
    RST = 0!
    RSR = 0!
    RTR = 0!
    ROM = 0!
    RMU = 0!
    K% = 4 * (I% - 1)         'COMPUTE THE SUB-INTEGRALS
    FOR J% = 1 TO 4
      J1% = K% + J%
      P = AZ(J1%, GNum%)
      EP = EXP(P * Z)
      T1 = B(J1%, L%) * EP
      T2 = D(J1%, L%) / EP
      T1P = T1 + T2
      T1M = T1 - T2
      T1 = (A(J1%, L%) + B(J1%, L%) * Z) * EP
      T2 = (C(J1%, L%) + D(J1%, L%) * Z) / EP
      T2P = P * (T1 + T2)
      T2M = P * (T1 - T2)
      WA = AJ(J1%, GNum%) * W(J%)
      IF RR(GNum%) > 0 THEN
        BJ1 = RJ1(J1%, GNum%) * P
        BJ0 = RJ0(J1%, GNum%) * P
        RSZ = RSZ + WA * P * BJ0 * (VL1 * T1P - T2M)
        ROM = ROM + WA * EL * BJ0 * (2! * VL1 * T1M - T2P)
        RTR = RTR + WA * P * BJ1 * (VL * T1M + T2P)
        RMU = RMU + WA * EL * BJ1 * (T1P + T2M)
        RSR = RSR + WA * (P * BJ0 * ((1! + VL) * T1P + T2M) - BJ1 * (T1P + T2M) / RR(GNum%))
        RST = RST + WA * (VL * P * BJ0 * T1P + BJ1 * (T1P + T2M) / RR(GNum%))
      ELSE                      'SPECIAL ROUTINE FOR R = ZERO
        PP = P * P
        RSZ = RSZ + WA * PP * (VL1 * T1P - T2M)
        ROM = ROM + WA * EL * P * (2! * VL1 * T1M - T2P)
        RST = RST + WA * PP * ((VL + .5) * T1P + .5 * T2M)
        RSR = RST
      END IF
    NEXT J%
    SF = (AZ(K% + 4, GNum%) - AZ(K% + 1, GNum%)) / 1.722273
    CSZ = CSZ + RSZ * SF
    CST = CST + RST * SF
    CSR = CSR + RSR * SF
    CTR = CTR + RTR * SF
    COMM = COMM + ROM * SF
    CMU = CMU + RMU * SF
    RSZ = 2! * RSZ * TLPR * SF
    TESTH = ABS(RSZ) - .0001
    IF ITS% < NTS1% THEN
      Test(ITS%) = TESTH
      ITS% = ITS% + 1
    ELSE
      Test(NTS1%) = TESTH
```

```
      FOR J% = 1 TO NTest%(GNum%)
        IF TESTH < Test(J%) THEN TESTH = Test(J%)
        Test(J%) = Test(J% + 1)
      NEXT J%
      IF TESTH <= 0 THEN EXIT FOR
    END IF
  NEXT I%
  CSZ = CSZ * ARP
  CST = CST * ARP
  CTR = CTR * ARP
  CSR = CSR * ARP
  COMM = COMM * ARP
  CMU = CMU * ARP
  BSTS = CSZ + CST + CSR
  BST = BSTS * (1! - 2! * V(L%)) / E(L%)
  IF TZZ > 0 THEN Z = -Z
  RDZ = (1000000! / E(L%)) * (CSZ - V(L%) * (CSR + CST))
  RDT = (1000000! / E(L%)) * (CST - V(L%) * (CSZ + CSR))
  RDS = (1000000! / E(L%)) * (CSR - V(L%) * (CSZ + CST))
  SST = (2000000! / E(L%)) * (1! + V(L%)) * CTR
'CALCULATE MAXIMUM PRINCIPAL STRAIN IN TENSILE DIRECTION AND ITS ANGLE OR
'ANGLES WITH SPATIAL AXES T,R, AND V.  IN PRINTOUT  A NUMERICAL ANGLE IS
'DIRECTION OF THIS STRAIN WITH R-AXIS IN THE R-V PLANE AND MINUS IS
'COUNTERCLOCKWISE.  IN PRINTOUT  A COMBINATION DIRECTION DEFINES THE PLANE OR
'PLANES IN WHICH THIS STRAIN IS CONSTANT.
  BSC = ABS(CTR) - .0009
  IF BSC <= 0! THEN       'WHEN SHEAR STRESS=0, T,R,& V ARE PRINCIPAL AXES.
    TMPMX1 = (1000000! / E(L%)) * (CSR - V(L%) * (CSZ + CST))
    TMPMX2 = (1000000! / E(L%)) * (CSZ - V(L%) * (CSR + CST))
    TMPMX3 = (1000000! / E(L%)) * (CST - V(L%) * (CSR + CSZ))
    MARY = (CST - CSR)
    THOMP = ABS(MARY) - .0009
    SUTTON = (CSR - CSZ)
    JUNIOR = ABS(SUTTON) - .0009
    SAM = (CST - CSZ)
    SAMPAT = ABS(SAM) - .0009
    IF JUNIOR < 0! AND THOMP < 0! THEN
      MAXSTR = TMPMX1
      CS1 = CSR
      CS3 = CST
    ELSEIF JUNIOR < 0! AND MARY <= -.0009 THEN
      MAXSTR = TMPMX1
      CS1 = CSR
      CS3 = CST
    ELSEIF SAMPAT < 0! AND MARY >= .0009 THEN
      MAXSTR = TMPMX2
      CS1 = CSZ
      CS3 = CSR
    ELSEIF THOMP < 0! AND SUTTON >= .0009 THEN
      MAXSTR = TMPMX3
      CS1 = CST
      CS3 = CSZ
    ELSEIF MARY >= .0009 AND SAM >= .0009 THEN
      MAXSTR = TMPMX3
      CS1 = CST
      CS3 = CSZ
    ELSEIF MARY <= -.0009 AND SUTTON >= .0009 THEN
      MAXSTR = TMPMX1
      CS1 = CSR
      CS3 = CST
    ELSE
      MAXSTR = TMPMX2
      CS1 = CSZ
      CS3 = CST
    END IF
  ELSE
'WHEN SHEAR STRESS IS NOT ZERO, PRINCIPAL STRAIN MAY BE AN ANGLE
'IN R-V PLANE OR IN T-DIRECTION OR MAY BE IN A COMBINATION (OF THESE) PLANE
    TEMP = .5 * (SQR((CSR - CSZ) * (CSR - CSZ) + 4! * CTR * CTR))
    CS1TEN = .5 * (CSZ + CSR) + TEMP
    CS2TEN = CS1TEN - 2! * TEMP
    CS3TEN = CST
    GEORGE = CS1TEN - CS3TEN
```

```
    TUT = ABS(GEORGE) - .0009
    IF GEORGE > -.0009 THEN
      MAN = 1000!
      COFF = CSR - CSZ
      IF COFF <= -.0009 THEN MAN = 0!
      SAM = ABS(COFF) - .0009
      IF SAM > 0 THEN     'THETA CAN VARY FROM +/- 0 TO 90 DEGREES WITH R-AXIS
        THETA = .5 * SIN(CTR / TEMP)
        THETA = (180! / Pi) * THETA
        IF MAN <= .000001 AND CTR >= .0009 THEN THETA = 90! - THETA
        IF MAN <= .000001 AND CTR <= -.0009 THEN THETA = -(90! - ABS(THETA))
      ELSE                'WHEN STRESSES IN R & V DIRECTIONS ARE EQUAL, THETA IS +/- 45 DEG WITH R-AXIS
        IF CTR >= .0009 THEN THETA = 45!
        IF CTR <= -.0009 THEN THETA = -45!
      END IF
      CS1 = CS1TEN
      IF CS3TEN - CS2TEN >= .0009 THEN
        CS2 = CS3TEN
        CS3 = CS2TEN
      ELSE
        CS2 = CS2TEN
        CS3 = CS3TEN
      END IF
    ELSE
      CS1 = CS3TEN
      CS2 = CS1TEN
      CS3 = CS2TEN
    END IF
  END IF
  IF NOpt% = 1 THEN CDEV = COMM
  IF NOpt% = 2 THEN CDEV = ABS(CS3 - CS1)
END SUB

SUB CHEVRON (WGT, GNum%, ZCOMP, NOpt%, CDEV) STATIC
'CHEVRON N-LAYER ELASTIC SYSTEMS PROGRAM
'CALCULATING STRESSES, STRAINS, AND DEFLECTIONS
'COMPUTE ZEROS OF J1(X) AND JO(X). SET UP GAUSS CONSTANTS

  REDIM Test(11), ZZ(1)

  PSI = WGT / (Pi * TLPR ^ 2!)
  ZZ(1) = ZCOMP
  FOR J% = 1 TO N%
    TZ = ABS(H(J%) - ZZ(1))
    IF TZ <= .0001 THEN ZZ(1) = -H(J%)
  NEXT J%
  IZT% = 1
  Z = ABS(ZZ(IZT%))           'FIND THE LAYER CONTAINING Z
  TZZ = 0!
  SELECT CASE NOpt%
    CASE 1
      L% = 1
    CASE 2
      L% = N%
  END SELECT
  CALL CALCIN(GNum%, CDEV, NOpt%, L%)
END SUB

SUB ChevronAnalysis (AnalysisFailed%) STATIC
  REDIM MeanTestLoad(NumStations%, MaxHeight%)
  REDIM MeanDefl(NumStations%, MaxHeight%, NumDeflectors%)
  REDIM ChevDefl(3, NumStations%, MaxHeight%, NumDeflectors%)
  REDIM ChevEsg(NumStations%, MaxHeight%, NumDeflectors%)
  REDIM DeviatorStress(NumStations%, MaxHeight%, NumDeflectors%)
  REDIM BackCalcEsg(NumStations%, MaxHeight%)
  REDIM EquivalentSN(NumStations%, MaxHeight%)
  REDIM RR(NumDeflectors%), Geo%(NumDeflectors%), NTest%(NumDeflectors%)
  REDIM E(12), V(12), H(11), BZ(50), AZ(184, 4), PZ(6), QZ(6), PO(6), QO(6)
  REDIM TPMeanTestLoad(NumTPStations%, MaxHeight%)
  REDIM TPMeanDefl(NumTPStations%, MaxHeight%, NumDeflectors%)
  REDIM TPChevDefl(3, NumTPStations%, MaxHeight%, NumDeflectors%)
  REDIM TPChevEsg(NumTPStations%, MaxHeight%, NumDeflectors%)
  REDIM TPDeviatorStress(NumTPStations%, MaxHeight%, NumDeflectors%)
```

```
  REDIM TPBackCalcEsg(NumTPStations%, MaxHeight%)
  REDIM TPEquivalentSN(NumTPStations%, MaxHeight%)
  REDIM Linearity%(3)

  AnalysisFailed% = False%

  CALL NormalColor
  LOCATE 25, 4: PRINT SPACE$(75);       'Clear Status Line
  LOCATE 10, 25: PRINT "Performing Elastic Layer Analysis"
  LOCATE 12, 28: PRINT "Percent Complete:      %"
  LOCATE 13, 28: PRINT "      Start Time: "; TIME$
  LOCATE 14, 28: PRINT "    Current Time: "
  CALL HiliteColor
  LOCATE 12, 46: PRINT USING "###.#"; 0
  LOCATE 14, 46: PRINT TIME$

  N1 = .35
  P1 = 1 - N1 ^ 2
  N2 = .45
  P2 = 1 - N2 ^ 2
  PZ(1) = 1!:            QZ(1) = -.005
  PZ(2) = -.0001125:     QZ(2) = 4.6875E-06
  PZ(3) = 2.871094E-07:  QZ(3) = -2.325586E-08
  PZ(4) = -2.344966E-09: QZ(4) = 2.830709E-10
  PZ(5) = 3.980684E-11:  QZ(5) = -6.39121E-12
  PZ(6) = -1.153613E-12: QZ(6) = 2.31247E-12
  P0(1) = 1!:            Q0(1) = .015
  P0(2) = .0001875:      Q0(2) = -6.5625E-06
  P0(3) = -3.691406E-07: Q0(3) = 2.842383E-08
  P0(4) = 2.771323E-09:  Q0(4) = -3.266202E-10
  P0(5) = -4.511442E-11: Q0(5) = 7.143117E-12
  P0(6) = 1.275046E-12:  Q0(6) = -2.532706E-13
  BZ(1) = 0!
  BZ(2) = 1!
  BZ(3) = 2.4048
  BZ(4) = 3.8317
  BZ(5) = 5.5201
  BZ(6) = 7.0156
  FOR I% = 7 TO 47 STEP 2
    T = I% \ 2
    TD = 4! * T - 1!
    BZ(I%) = Pi * (T - .25 + (.050661 / TD) - (.053041 / TD ^ 3) + (.262051 / TD ^ 5))
  NEXT I%
  FOR I% = 8 TO 46 STEP 2
    T = (I% - 2) / 2
    TD = 4! * T + 1!
    BZ(I%) = Pi * (T + .25 - (.151982 / TD) + (.015399 / TD ^ 3) - (.24527 / TD ^ 5))
  NEXT I%

' Calc Mean Test Load and Defl:
  FOR I% = 1 TO NumStations%
    REDIM Sum(NumDeflectors%)
    SumLoad = 0
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF Defl(I%, J%, 1) > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            Sum(K%) = Sum(K%) + Defl(I%, J%, K%)
          NEXT K%
          SumLoad = SumLoad + TestLoad(I%, J%)
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            MeanDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
            Sum(K%) = Defl(I%, J%, K%)
          NEXT K%
          MeanTestLoad(I%, LastHeight%) = SumLoad / ACount%
          SumLoad = TestLoad(I%, J%)
        END IF
```

```
        LastHeight% = Heights%(J%)
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            MeanDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
          NEXT K%
          MeanTestLoad(I%, LastHeight%) = SumLoad / ACount%
        END IF
      END IF
    NEXT J%
  NEXT I%
' Calc Mean Test Load and Defl for test pits
  FOR I% = 1 TO NumTPStations%
    REDIM Sum(NumDeflectors%)
    SumLoad = 0
    ACount% = 0
    FOR J% = 1 TO InitNumPeaks%
      IF J% = 1 THEN LastHeight% = Heights%(J%)
      IF Heights%(J%) = LastHeight% THEN
        IF TPDefl(I%, J%, 1) > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            Sum(K%) = Sum(K%) + TPDefl(I%, J%, K%)
          NEXT K%
          SumLoad = SumLoad + TPTestLoad(I%, J%)
          ACount% = ACount% + 1
        END IF
      ELSE
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            TPMeanDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
            Sum(K%) = TPDefl(I%, J%, K%)
          NEXT K%
          TPMeanTestLoad(I%, LastHeight%) = SumLoad / ACount%
          SumLoad = TPTestLoad(I%, J%)
        END IF
        LastHeight% = Heights%(J%)
        ACount% = 1
      END IF
      IF J% = InitNumPeaks% THEN
        IF ACount% > 0 THEN
          FOR K% = 1 TO NumDeflectors%
            TPMeanDefl(I%, LastHeight%, K%) = Sum(K%) / ACount%
          NEXT K%
          TPMeanTestLoad(I%, LastHeight%) = SumLoad / ACount%
        END IF
      END IF
    NEXT J%
  NEXT I%
  ERASE Sum

' Determine Equivalent Radius of Contact at Subgrade Interface:
  ESG = 5: HE = 0
  FOR I% = 1 TO NumLayers%
    HE = HE + .9 * Thickness(I%) * (Modulus(I%) / ESG * N2 / (1 - PoissonRatio(I%) ^ 2)) ^ (1 / 3)
  NEXT I%
  ae = TLPR + HE
  FOR I% = 1 TO NumDeflectors%
    IF RadFactor * Radial(I%) > ae THEN StartDeflector% = I%: EXIT FOR
  NEXT I%
  IF StartDeflector% = 0 THEN StartDeflector% = NumDeflectors% - 2
  IF (NumDeflectors% - StartDeflector%) < 2 THEN StartDeflector% = NumDeflectors% - 2

  DCnt% = 1
  RR(DCnt%) = Radial(1) * RadFactor
  Geo%(DCnt%) = 1
  FOR I% = StartDeflector% TO NumDeflectors%
    DCnt% = DCnt% + 1
    RR(DCnt%) = Radial(I%) * RadFactor
    Geo%(DCnt%) = I%
  NEXT I%
  REDIM AJ(184, DCnt%), RJ1(184, DCnt%), RJ0(184, DCnt%)
```

```
  CALL Part(DCnt%)                    'calculate partition
  FOR GNum% = 1 TO DCnt%
    FOR I% = 1 TO 184            'CALCULATE THE COEFFICIENTS
      IF RR(GNum%) > 0 THEN
        PR = AZ(I%, GNum%) * RR(GNum%)
        CALL BESSEL(0, PR, RJ0(I%, GNum%))
        CALL BESSEL(1, PR, RJ1(I%, GNum%))
      END IF
      PA = AZ(I%, GNum%) * TLPR
      CALL BESSEL(1, PA, AJ(I%, GNum%))
    NEXT I%
  NEXT GNum%

' Perform Chevron Analysis for Esg = 5000, 15000, 30000:
  ZInterface = 0
  FOR I% = 1 TO NumLayers%
    E(I%) = Modulus(I%) * 1000
    V(I%) = PoissonRatio(I%)
    ZInterface = ZInterface + Thickness(I%)
    H(I%) = ZInterface                'depth to bottom of layer
  NEXT I%
  TotalThick = ZInterface
  V(NumLayers% + 1) = .4
  V(NumLayers% + 2) = .35
  E(NumLayers% + 2) = 1000000
  H(NumLayers% + 1) = DepthRigidLayer
  NS% = NumLayers% + 2                'for CHEVRON common block
  N% = NS% - 1
  Opt% = 1                            'calculate deflections
  ZCOMP = 0
  ZCount% = 0
  ZTotal% = 3 * (NumStations% + NumTPStations%) * (DCnt% - 1)
  FOR II% = 1 TO 3
    SELECT CASE II%
      CASE 1
        E(NumLayers% + 1) = 5000
      CASE 2
        E(NumLayers% + 1) = 15000
      CASE 3
        E(NumLayers% + 1) = 30000
    END SELECT
    FOR GNum% = 2 TO DCnt%            '2nd to last deflector in list
      REDIM A(184, 10), B(184, 10), C(184, 10), D(184, 10), XX(10, 4, 4), SC(9), FM(4), PM(9, 4, 4)
      FOR JK% = 1 TO 184              'CALCULATE THE COEFFICIENTS
        CALL COFE(JK%, AZ(JK%, GNum%))
      NEXT JK%
      FOR JJ% = 1 TO NumStations%
        FOR KK% = 1 TO MaxHeight%
          IF HeightList%(KK%) THEN
            MeanLoad = MeanTestLoad(JJ%, KK%) * LoadFactor
            CALL CHEVRON(MeanLoad, GNum%, ZCOMP, Opt%, DEFDEV)
            IF DEFDEV > 0 THEN
              ChevDefl(II%, JJ%, KK%, Geo%(GNum%)) = DEFDEV
              LOCATE 14, 46: PRINT TIME$
            ELSE
              REDIM PUText$(5)
              IF DepthRigidLayer < 1200 THEN
                PUText$(1) = "Chevron analysis failed, possibly due to shallow rigid layer at" + STR$(DepthRigidLayer / 12)
+ " feet."
              ELSE
                PUText$(1) = "Chevron analysis failed due to an unknown cause at:"
              END IF
              PUText$(2) = "            Station" + STR$(Station%(JJ%)) + " feet"
              PUText$(3) = "             Height" + STR$(KK%)
              PUText$(4) = "           Geophone" + STR$(Geo%(GNum%))
              PUText$(5) = "     Subgrade trial" + STR$(II%)
              CALL PopupError
              AnalysisFailed% = True%
              GOTO CleanUP
            END IF
          END IF
        NEXT KK%
        ZCount% = ZCount% + 1
```

```
        LOCATE 12, 46: PRINT USING "###.#"; (29 * (ZCount% / ZTotal%))
      NEXT JJ%
      FOR JJ% = 1 TO NumTPStations%
        FOR KK% = 1 TO MaxHeight%
          IF HeightList%(KK%) THEN
            MeanLoad = TPMeanTestLoad(JJ%, KK%) * LoadFactor
            CALL CHEVRON(MeanLoad, GNum%, ZCOMP, Opt%, DEFDEV)
            IF DEFDEV > 0 THEN
              TPChevDefl(II%, JJ%, KK%, Geo%(GNum%)) = DEFDEV
              LOCATE 14, 46: PRINT TIME$
            ELSE
              REDIM PUText$(5)
              IF DepthRigidLayer < 1200 THEN
                PUText$(1) = "Chevron analysis failed, possibly due to shallow rigid layer at" + STR$(DepthRigidLayer / 12)
+ " feet."
              ELSE
                PUText$(1) = "Chevron analysis failed due to an unknown cause at:"
              END IF
              PUText$(2) = "            Station " + LTRIM$(STR$(TPStation%(JJ%))) + " feet"
              PUText$(3) = "             Height" + STR$(KK%)
              PUText$(4) = "           Geophone" + STR$(Geo%(GNum%))
              PUText$(5) = "     Subgrade trial" + STR$(II%)
              CALL PopupError
              AnalysisFailed% = True%
              GOTO CleanUP
            END IF
          END IF
        NEXT KK%
        ZCount% = ZCount% + 1
        LOCATE 12, 46: PRINT USING "###.#"; (29 * (ZCount% / ZTotal%))
      NEXT JJ%
    NEXT GNum%
  NEXT II%

' Determine Esg for Deflectors: (uncorrected deflections)
  FOR J% = 1 TO NumStations%
    FOR K% = 1 TO MaxHeight%
      IF HeightList%(K%) THEN
        FOR L% = StartDeflector% TO NumDeflectors%
          IF MeanDefl(J%, K%, L%) * DeflFactor > ChevDefl(2, J%, K%, L%) THEN
            Slope = Log10(ChevDefl(2, J%, K%, L%) / ChevDefl(1, J%, K%, L%)) / Log10(15000 / 5000)
            IF Slope <> 0 THEN
              ChevEsg(J%, K%, L%) = 10 ^ ((Log10(MeanDefl(J%, K%, L%) * DeflFactor) - Log10(ChevDefl(1, J%, K%, L%))) / Slope
+ Log10(5000))
            END IF
          ELSE
            Slope = Log10(ChevDefl(3, J%, K%, L%) / ChevDefl(2, J%, K%, L%)) / Log10(30000 / 15000)
            IF Slope <> 0 THEN
              ChevEsg(J%, K%, L%) = 10 ^ ((Log10(MeanDefl(J%, K%, L%) * DeflFactor) - Log10(ChevDefl(2, J%, K%, L%))) / Slope
+ Log10(15000))
            END IF
          END IF
          LOCATE 14, 46: PRINT TIME$
        NEXT L%
      END IF
    NEXT K%
  NEXT J%
  FOR J% = 1 TO NumTPStations%
    FOR K% = 1 TO MaxHeight%
      IF HeightList%(K%) THEN
        FOR L% = StartDeflector% TO NumDeflectors%
          IF TPMeanDefl(J%, K%, L%) * DeflFactor > TPChevDefl(2, J%, K%, L%) THEN
            Slope = Log10(TPChevDefl(2, J%, K%, L%) / TPChevDefl(1, J%, K%, L%)) / Log10(15000 / 5000)
            IF Slope <> 0 THEN
              TPChevEsg(J%, K%, L%) = 10 ^ ((Log10(TPMeanDefl(J%, K%, L%) * DeflFactor) - Log10(TPChevDefl(1, J%, K%, L%)))
/ Slope + Log10(5000))
            END IF
          ELSE
            Slope = Log10(TPChevDefl(3, J%, K%, L%) / TPChevDefl(2, J%, K%, L%)) / Log10(30000 / 15000)
            IF Slope <> 0 THEN
              TPChevEsg(J%, K%, L%) = 10 ^ ((Log10(TPMeanDefl(J%, K%, L%) * DeflFactor) - Log10(TPChevDefl(2, J%, K%, L%)))
/ Slope + Log10(15000))
            END IF
```

```
        END IF
        LOCATE 14, 46: PRINT TIME$
      NEXT L%
    END IF
  NEXT K%
NEXT J%
LOCATE 12, 46: PRINT USING "###.#"; 30

' Perform Chevron Analysis for Determined Esg to find Deviator Stresses:
ZCOMP = ZInterface
Opt% = 2
NS% = NumLayers% + 2                    'for CHEVRON common block
N% = NS% - 1
ZCount% = 0
ZTotal% = NumStations% + NumTPStations%
FOR J% = 1 TO NumStations%
  FOR K% = 1 TO MaxHeight%
    IF HeightList%(K%) THEN
      FOR GNum% = 1 TO DCnt%
        IF GNum% = 1 THEN
          E(NumLayers% + 1) = ChevEsg(J%, K%, StartDeflector%)
        ELSE
          E(NumLayers% + 1) = ChevEsg(J%, K%, Geo%(GNum%))
        END IF
        REDIM A(184, 10), B(184, 10), C(184, 10), D(184, 10), XX(10, 4, 4), SC(9), FM(4), PM(9, 4, 4)
        FOR JK% = 1 TO 184              'CALCULATE THE COEFFICIENTS
          CALL COFE(JK%, AZ(JK%, GNum%))
        NEXT JK%
        MeanLoad = MeanTestLoad(J%, K%) * LoadFactor
        CALL CHEVRON(MeanLoad, GNum%, ZCOMP, Opt%, DEFDEV)
        DeviatorStress(J%, K%, Geo%(GNum%)) = DEFDEV
        LOCATE 14, 46: PRINT TIME$
      NEXT GNum%
    END IF
  NEXT K%
  ZCount% = ZCount% + 1
  LOCATE 12, 46: PRINT USING "###.#"; (30 + 65 * (ZCount% / ZTotal%))
NEXT J%
FOR J% = 1 TO NumTPStations%
  FOR K% = 1 TO MaxHeight%
    IF HeightList%(K%) THEN
      FOR GNum% = 1 TO DCnt%
        IF GNum% = 1 THEN
          E(NumLayers% + 1) = TPChevEsg(J%, K%, StartDeflector%)
        ELSE
          E(NumLayers% + 1) = TPChevEsg(J%, K%, Geo%(GNum%))
        END IF
        REDIM A(184, 10), B(184, 10), C(184, 10), D(184, 10), XX(10, 4, 4), SC(9), FM(4), PM(9, 4, 4)
        FOR JK% = 1 TO 184              'CALCULATE THE COEFFICIENTS
          CALL COFE(JK%, AZ(JK%, GNum%))
        NEXT JK%
        MeanLoad = TPMeanTestLoad(J%, K%) * LoadFactor
        CALL CHEVRON(MeanLoad, GNum%, ZCOMP, Opt%, DEFDEV)
        TPDeviatorStress(J%, K%, Geo%(GNum%)) = DEFDEV
        LOCATE 14, 46: PRINT TIME$
      NEXT GNum%
    END IF
  NEXT K%
  ZCount% = ZCount% + 1
  LOCATE 12, 46: PRINT USING "###.#"; (30 + 65 * (ZCount% / ZTotal%))
NEXT J%

' Determine Esg:
Num% = NumDeflectors% - StartDeflector% + 1
Conn% = NumDeflectors% - Num%
ZCount% = 0
ZTotal% = NumStations% + NumTPStations%
REDIM xxx(Num%), yyy(Num%)
FOR J% = 1 TO NumStations%
  FOR K% = 1 TO MaxHeight%
    IF HeightList%(K%) THEN
      MinEsg = 1E+20
      FOR aaa% = 1 TO Num%
```

```
        xxx(aaa%) = Log10(DeviatorStress(J%, K%, aaa% + Conn%))
        yyy(aaa%) = Log10(ChevEsg(J%, K%, aaa% + Conn%))
        IF ChevEsg(J%, K%, aaa% + Conn%) < MinEsg THEN MinEsg = ChevEsg(J%, K%, aaa% + Conn%)
      NEXT aaa%
      CALL LinearLeastSquaresFit(Num%, xxx(), yyy(), Constant, Slope, Rsqd)
      BackCalcEsg(J%, K%) = 10 ^ (Log10(DeviatorStress(J%, K%, 1)) * Slope + Constant)
      SELECT CASE Slope
        CASE IS > -.15              'linear response
          Linearity%(1) = Linearity%(1) + 1
        CASE IS > -.4               'slightly non-linear response
          Linearity%(2) = Linearity%(2) + 1
        CASE ELSE                   'distinctly non-linear response
          Linearity%(3) = Linearity%(3) + 1
      END SELECT
      IF MinEsg < BackCalcEsg(J%, K%) THEN BackCalcEsg(J%, K%) = MinEsg
      LOCATE 14, 46: PRINT TIME$
    END IF
  NEXT K%
  ZCount% = ZCount% + 1
  LOCATE 12, 46: PRINT USING "###.#"; 95 + 2 * (ZCount% / ZTotal%)
NEXT J%
FOR J% = 1 TO NumTPStations%
  FOR K% = 1 TO MaxHeight%
    IF HeightList%(K%) THEN
      MinEsg = 1E+20
      FOR aaa% = 1 TO Num%
        xxx(aaa%) = Log10(TPDeviatorStress(J%, K%, aaa% + Conn%))
        yyy(aaa%) = Log10(TPChevEsg(J%, K%, aaa% + Conn%))
        IF TPChevEsg(J%, K%, aaa% + Conn%) < MinEsg THEN MinEsg = TPChevEsg(J%, K%, aaa% + Conn%)
      NEXT aaa%
      CALL LinearLeastSquaresFit(Num%, xxx(), yyy(), Constant, Slope, Rsqd)
      TPBackCalcEsg(J%, K%) = 10 ^ (Log10(TPDeviatorStress(J%, K%, 1)) * Slope + Constant)
      SELECT CASE Slope
        CASE IS > -.15              'linear response
          Linearity%(1) = Linearity%(1) + 1
        CASE IS > -.4               'slightly non-linear response
          Linearity%(2) = Linearity%(2) + 1
        CASE ELSE                   'distinctly non-linear response
          Linearity%(3) = Linearity%(3) + 1
      END SELECT
      IF MinEsg < TPBackCalcEsg(J%, K%) THEN TPBackCalcEsg(J%, K%) = MinEsg
      LOCATE 14, 46: PRINT TIME$
    END IF
  NEXT K%
  ZCount% = ZCount% + 1
  LOCATE 12, 46: PRINT USING "###.#"; 95 + 2 * (ZCount% / ZTotal%)
NEXT J%

' Determine EquivalentSN:
ZCount% = 0
ZTotal% = NumStations% + NumTPStations%
FOR J% = 1 TO NumStations%
  FOR K% = 1 TO MaxHeight%
    IF HeightList%(K%) THEN
      MeanLoad = MeanTestLoad(J%, K%) * LoadFactor
      PreDef = MeanCorrDefl(J%, K%) * DeflFactor * MeanLoad
      K1 = 2 * MeanLoad * P2 / (Pi * TLPR * BackCalcEsg(J%, K%))
      SN = .3
      DO
        EE = ((SN / (.0043 * TotalThick)) ^ 3) * P2
        HE = .9 * TotalThick * (((EE * P2) / (BackCalcEsg(J%, K%) * P1)) ^ (1 / 3))
        K2 = HE / TLPR
        K3 = SQR(1 + K2 ^ 2) - K2
        K4 = 2 * (1 - N2) * (SQR(1 + K2 ^ 2))
        FB = K3 * (1 + K2 / K4)
        K5 = ((BackCalcEsg(J%, K%) * P1) / (EE * P2))
        K6 = 1 - (BackCalcEsg(J%, K%) / EE)
        FW = K5 + FB * K6
        DR0 = K1 * FW
        K7 = ((PreDef - DR0) / PreDef) * 100!
        K8 = SQR(K7 ^ 2)
        IF DR0 < PreDef OR K8 < 1! OR SN >= 20 THEN EXIT DO
        SN = SN + .05
```

```
        LOOP
        EquivalentSN(J%, K%) = SN
        LOCATE 14, 46: PRINT TIME$
      END IF
    NEXT K%
    ZCount% = ZCount% + 1
    LOCATE 12, 46: PRINT USING "###.#"; 97 + 3 * (ZCount% / ZTotal%)
  NEXT J%
  FOR J% = 1 TO NumTPStations%
    FOR K% = 1 TO MaxHeight%
      IF HeightList%(K%) THEN
        MeanLoad = TPMeanTestLoad(J%, K%) * LoadFactor
        PreDef = TPMeanCorrDefl(J%, K%) * DeflFactor * MeanLoad
        K1 = 2 * MeanLoad * P2 / (Pi * TLPR * TPBackCalcEsg(J%, K%))
        SN = .3
        DO
          EE = ((SN / (.0043 * TotalThick)) ^ 3) * P2
          HE = .9 * TotalThick * (((EE * P2) / (TPBackCalcEsg(J%, K%) * P1)) ^ (1 / 3))
          K2 = HE / TLPR
          K3 = SQR(1 + K2 ^ 2) - K2
          K4 = 2 * (1 - N2) * (SQR(1 + K2 ^ 2))
          FB = K3 * (1 + K2 / K4)
          K5 = ((TPBackCalcEsg(J%, K%) * P1) / (EE * P2))
          K6 = 1 - (TPBackCalcEsg(J%, K%) / EE)
          FW = K5 + FB * K6
          DRO = K1 * FW
          K7 = ((PreDef - DRO) / PreDef) * 100!
          K8 = SQR(K7 ^ 2)
          IF DRO < PreDef OR K8 < 1! OR SN >= 20 THEN EXIT DO
          SN = SN + .05
        LOOP
        TPEquivalentSN(J%, K%) = SN
        LOCATE 14, 46: PRINT TIME$
      END IF
    NEXT K%
    ZCount% = ZCount% + 1
    LOCATE 12, 46: PRINT USING "###.#"; 97 + 3 * (ZCount% / ZTotal%)
  NEXT J%

  LoSN = 0: HiSN = 0
  FOR I% = 1 TO NumLayers%
    LoSN = LoSN + Thickness(I%) * ALo(I%)
    HiSN = HiSN + Thickness(I%) * AHi(I%)
  NEXT I%
  AvgSN = (HiSN + LoSN) / 2

CleanUP:
  ERASE ChevDefl, ChevEsg, DeviatorStress, MeanTestLoad, MeanDefl
  ERASE TPChevDefl, TPChevEsg, TPDeviatorStress, TPMeanTestLoad, TPMeanDefl
  ERASE E, V, H, PZ, QZ, P0, Q0
END SUB

SUB COFE (LC%, P) STATIC
'USE FOR ALL PROBLEMS UP TO MAX DIMENSION OF 15 LAYERS
'REPROGRAMMED 1 MAY 1980 BY L J PAINTER
'NOTE DOUBLE ENTRY POINT FOR COE5 AND CO15
  REDIM Q(2, 2)
  FOR K% = 1 TO N%
    T1 = E(K%) * (1! + V(K% + 1)) / (E(K% + 1) * (1! + V(K%)))
    T1M = T1 - 1!
    PH = P * H(K%)
    PH2 = PH * 2!
    VK2 = 2! * V(K%)
    VKP2 = 2! * V(K% + 1)
    VK4 = 2! * VK2
    VKP4 = 2! * VKP2
    VKK8 = 8! * V(K%) * V(K% + 1)
    XX(K%, 1, 1) = VK4 - 3! - T1
    XX(K%, 2, 1) = 0!
    XX(K%, 3, 1) = T1M * (PH2 - VK4 + 1!)
    XX(K%, 4, 1) = -2! * T1M * P
    T3 = PH2 * (VK2 - 1!)
    T4 = VKK8 + 1! - 3! * VKP2
```

```
    T5 = PH2 * (VKP2 - 1!)
    T6 = VKK8 + 1! - 3! * VK2
    XX(K%, 1, 2) = (T3 + T4 - T1 * (T5 + T6)) / P
    XX(K%, 2, 2) = T1 * (VKP4 - 3!) - 1!
    XX(K%, 4, 2) = T1M * (1! - PH2 - VKP4)
    XX(K%, 3, 4) = (T3 - T4 - T1 * (T5 - T6)) / P
    T3 = PH2 * PH - VKK8 + 1!
    T4 = PH2 * (VK2 - VKP2)
    XX(K%, 1, 4) = (T3 + T4 + VKP2 - T1 * (T3 + T4 + VK2)) / P
    XX(K%, 3, 2) = (-T3 + T4 - VKP2 + T1 * (T3 - T4 + VK2)) / P
    XX(K%, 1, 3) = T1M * (1! - PH2 - VK4)
    XX(K%, 2, 3) = 2! * T1M * P
    XX(K%, 3, 3) = VK4 - 3! - T1
    XX(K%, 4, 3) = 0!
    XX(K%, 2, 4) = T1M * (PH2 - VKP4 + 1!)
    XX(K%, 4, 4) = T1 * (VKP4 - 3!) - 1!
  NEXT K%
  SC(N%) = 4! * (V(N%) - 1!)
  IF N% >= 2 THEN
    FOR K1% = 2 TO N%
      M% = NS% - K1%
      SC(M%) = SC(M% + 1) * 4! * (V(M%) - 1!)
    NEXT K1%
  END IF
  Q(1, 1) = 1!
  Q(2, 2) = 1!
  Q(1, 2) = 0!
  QQ = P * 2! * H(N%)
  IF QQ <= 50 THEN Q(1, 2) = EXP(-QQ)
  FOR M% = 1 TO 4
    LL% = (M% + 1) \ 2
    FOR J% = 3 TO 4
      PM(N%, M%, J%) = XX(N%, M%, J%) * Q(LL%, 2)
    NEXT J%
  NEXT M%
  FOR K1% = 2 TO N%
    K% = NS% - K1%
    KK% = K% + 1
    QQ = P * 2! * H(K%)
    IF QQ <= 50 THEN
      Q(2, 1) = EXP(QQ)
      Q(1, 2) = 1! / Q(2, 1)
    ELSE
      Q(1, 2) = 0!
      Q(2, 1) = 1E+20
    END IF
    FOR M% = 1 TO 4
      LL% = (M% + 1) \ 2
      FOR J% = 3 TO 4
        PM(K%, M%, J%) = (XX(K%, M%, 1) * PM(KK%, 1, J%) + XX(K%, M%, 2) * PM(KK%, 2, J%)) * Q(LL%, 1) + (XX(K%, M%, 3) *
PM(KK%, 3, J%) + XX(K%, M%, 4) * PM(KK%, 4, J%)) * Q(LL%, 2)
      NEXT J%
    NEXT M%
  NEXT K1%
  T3 = 2! * V(1)
  T4 = T3 - 1!
  FM(1) = P * (PM(1, 1, 3) + PM(1, 3, 3)) + T3 * (PM(1, 2, 3) - PM(1, 4, 3))
  FM(2) = P * (PM(1, 1, 3) - PM(1, 3, 3)) + T4 * (PM(1, 2, 3) + PM(1, 4, 3))
  FM(3) = P * (PM(1, 1, 4) + PM(1, 3, 4)) + T3 * (PM(1, 2, 4) - PM(1, 4, 4))
  FM(4) = P * (PM(1, 1, 4) - PM(1, 3, 4)) + T4 * (PM(1, 2, 4) + PM(1, 4, 4))
  DFAC = SC(1) / ((FM(1) * FM(4) - FM(3) * FM(2)) * P * P)
  A(LC%, NS%) = 0!
  B(LC%, NS%) = 0!
  C(LC%, NS%) = -FM(3) * DFAC
  D(LC%, NS%) = FM(1) * DFAC
  FOR K1% = 1 TO N%
    A(LC%, K1%) = (PM(K1%, 1, 3) * C(LC%, NS%) + PM(K1%, 1, 4) * D(LC%, NS%)) / SC(K1%)
    B(LC%, K1%) = (PM(K1%, 2, 3) * C(LC%, NS%) + PM(K1%, 2, 4) * D(LC%, NS%)) / SC(K1%)
    C(LC%, K1%) = (PM(K1%, 3, 3) * C(LC%, NS%) + PM(K1%, 3, 4) * D(LC%, NS%)) / SC(K1%)
    D(LC%, K1%) = (PM(K1%, 4, 3) * C(LC%, NS%) + PM(K1%, 4, 4) * D(LC%, NS%)) / SC(K1%)
  NEXT K1%
END SUB
```

```
SUB Part (Ct%) STATIC
  G1 = .8611363
  G2 = .339981
  FOR GNum% = 1 TO Ct%
    ZF = TLPR
    NTest%(GNum%) = 2
    IF RR(GNum%) > 0 THEN
      NTest%(GNum%) = TLPR / RR(GNum%) + .0001
      IF NTest%(GNum%) <= 0 THEN
        NTest%(GNum%) = RR(GNum%) / TLPR + .0001
        ZF = RR(GNum%)
      END IF
      NTest%(GNum%) = NTest%(GNum%) + 1
      IF NTest%(GNum%) > 10 THEN NTest%(GNum%) = 10
    END IF
    K% = 1              'COMPUTE POINTS FOR LEGENDRE-GAUSS INTEGRATION
    ZF = 2! * ZF
    SZ2 = 0!
    FOR I% = 1 TO 46
      SZ1 = SZ2
      SZ2 = BZ(I% + 1) / ZF
      SF = SZ2 - SZ1
      PP = SZ2 + SZ1
      SG1 = SF * G1
      SG2 = SF * G2
      AZ(K%, GNum%) = PP - SG1
      AZ(K% + 1, GNum%) = PP - SG2
      AZ(K% + 2, GNum%) = PP + SG2
      AZ(K% + 3, GNum%) = PP + SG1
      K% = K% + 4
    NEXT I%
  NEXT GNum%
END SUB
```

Long-Term Pavement Performance Advisory Committee

Chairman
William J. MacCreery
W.J. MacCreery, Inc.

David Albright
Alliance for Transportation Research

Richard Barksdale
Georgia Institute of Technology

James L. Brown
Pavement Consultant

Robert L. Clevenger
Colorado Department of Highways

Ronald Collins
Georgia Department of Transportation

Guy Dore
Ministere des Transports de Quebec

Charles E. Dougan
Connecticut Department of Transportation

McRaney Fulmer
South Carolina Department of Highways and Public Transportation

Marlin J. Knutson
American Concrete Pavement Association

Hans Jorgen Ertman Larsen
Danish Road Institute, Road Directorate

Kenneth H. McGhee
Consultant Civil Engineer

Raymond K. Moore
University of Kansas

Richard D. Morgan
National Asphalt Pavement Association

William R. Moyer
Pennsylvania Department of Transportation

David E. Newcomb
University of Minnesota

Charles A. Pryor
National Stone Association

Cesar A.V. Queiroz
The World Bank

Roland L. Rizenbergs
Kentucky Transportation Cabinet

Gary K. Robinson
Arizona Department of Transportation

Frederic R. Ross
Wisconsin Department of Transportation

Ted M. Scott
American Trucking Association

Marshall R. Thompson
University of Illinois

Kenneth R. Wardlaw
Exxon Chemical Corporation

Marcus Williams
H.B. Zachry Company

Liaisons

Albert J. Bush, III
USAE Waterways Experiment Station

Louis M. Papet
Federal Highway Administration

John P. Hallin
Federal Highway Administration

Ted Ferragut
Federal Highway Administration

Frank R. McCullagh
Transportation Research Board

Expert Task Group

Paul D. Anderson
Mountainview Geotechnical Ltd.

Robert C. Briggs
Engineer of Pavement Management

Albert J. Bush, III
USAE Waterways Experimental Station

Billy G. Connor
Alaska Department of Transportation

William Edwards
Engineer Research and Development

John P. Hallin
Federal Highway Administration

Frank L. Holman, Jr.
Alabama Highway Department

William J. Kenis
Federal Highway Administration

Joe P. Mahoney
University of Washington

Larry A. Scofield
Arizona Transportation Research Center

Richard N. Stubstad
Dynatest Consulting, Inc.

Marshall R. Thompson
University of Illinois

Per Ullidtz
Technical University of Denmark

Jacob Uzan
Texas A&M University

Wes Yang
New York State Department of Transportation