BNL-61827
INFORMAL

# BASIC PROGRAMS FOR THE $CO_2$ SYSTEM IN SEAWATER

E. R. Lewis
D.W.R. Wallace

May 1995

OCEANOGRAPHIC AND ATMOSPHERIC SCIENCES DIVISION
DEPARTMENT OF APPLIED SCIENCE
BROOKHAVEN NATIONAL LABORATORY
UPTON, NY 11973

UNDER CONTRACT NO. DE-AC02-76CH00016 WITH THE
UNITED STATES DEPARTMENT OF ENERGY

DISCLAIMER



Printed in the United States of America
Available From
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

NTIS price codes:
Printed Copy: A06; Microfiche Copy: A01

## DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

TABLE OF CONTENTS

**Page**

BASIC PROGRAMS FOR THE $CO_2$ SYSTEM IN SEAWATER . . . . . 1

ABOUT THE PROGRAMS . . . . . 2

FURTHER INFORMATION . . . . . 4

REFERENCES . . . . . 5

ACKNOWLEDGEMENTS . . . . . 5

APPENDIX - Source Code Listing for CO2SYSTEM.BAS . . . . . A1

CO2SYSTM.WP5, version 1.02, 5-9-95

# BASIC PROGRAMS FOR THE $CO_2$ SYSTEM IN SEAWATER

The $CO_2$ system in seawater is characterized by four measurable parameters: TA, the total alkalinity; TC, the total (inorganic) $CO_2$ (the sum of the dissolved $CO_2$, the carbonate, and the bicarbonate); pH; and either $fCO_2$, the fugacity of $CO_2$, or $xCO_2$, the mole fraction of $CO_2$ in air. TA and TC are independent of temperature; $fCO_2$, $xCO_2$, and pH are not. The knowledge of any two of these parameters, along with the temperature, the salinity, the abundances of other constituents of seawater, and the relevant equilibrium constants, allows the determination of the other two.

$xCO_2$ is the mole fraction of $CO_2$ in air which is in equilibrium with the seawater. The partial pressure of $CO_2$ is the mole fraction times the total pressure (normally taken to be one atmosphere). The fugacity refers to the 'escaping-tendency' of a gas (similar in concept to the chemical potential of a species). This quantity is closely related to the partial pressure of a gas but takes into account the nonideality of the $CO_2$ in air mixture. The fugacity is about .3% to .4% lower than the partial pressure over the range of interest. Both $fCO_2$ and $xCO_2$ are proportional to the dissolved $CO_2$.

In these programs, the $xCO_2$ is assumed to be for dry air, while the $fCO_2$ is assumed to refer to wet (i.e. $H_2O$-saturated) air at 1 atmosphere total pressure. The vapor pressure of water in seawater accounts for some of this total pressure, so values in wet air will be smaller than values in dry air by an amount proportional to the vapor pressure of water, which is of the order of a few percent. At 10 deg C the vapor pressure of $H_2O$ is .0121 atm, while that of seawater with salinity 35 is .0119 atm. At 25 deg C the value for $H_2O$ is .0312 atm, while that of seawater is .0307 atm. Various equations for this quantity exist in the literature, but they give similar results. In these programs, the equation of Weiss and Price (1980) is used.

The other dissolved constituents of seawater which are important for the speciation of the $CO_2$ system are of two types: those which are conservative or proportional to the salinity (these include boron and sulfur species, for instance), and those which are non-conservative, particularly the nutrients. The nutrients which are important for this discussion are phosphate and silicate. Often their concentrations are small and their importance slight, though it is advisable that they be included if their concentrations are known. The effect of other constituents, such as magnesium, are incorporated in the definition of the equilibrium constants. Any artificial seawater medium should therefore have such elements present in ratios close to those found in seawater.

The relevant equilibrium constants which define the speciation of $CO_2$ in seawater have been determined for various temperatures and salinities by several different investigators. The most important constants are $K_0$, the solubility of $CO_2$ in seawater, and $K_1$ and $K_2$, the equilibrium constants for bicarbonate and carbonate. $K_0$ comes from Weiss (1974) who combined the measurements of Murray and Riley (1971) with some of his own and fit the

resulting data. Estimates of its accuracy vary from .2% (Weiss, 1974) to .5% (Dickson and Riley, 1978). For $K_1$ and $K_2$, four sets of measurements have been made which remain worthy of consideration. These were made by Roy et al. (1993), Goyet and Poisson (1989), Hansson (1973), and Mehrbach et al. (1973). The data of Hansson and Mehrbach et al have been refit by various authors; the differences in these fits do not differ greatly. Various pH scales have been used in determining the constants, four of which are found in discussions of seawater pH. These are the NBS scale, the free scale, the total scale, and the seawater scale (the latter referred to by some authors originally as the total seawater scale). These various pH scales are a source of confusion and misunderstanding. All of the measurements and calculations in these programs are based on the total scale. Various units of concentration are also used. The most common are molality (mol/kg-$H_2O$) and mol/kg-soln (the solution being seawater or an artificial medium approximating seawater). In some cases, constants have been reported in molarity (mol/liter), but this is not common anymore. All constants used in these programs have been converted to units of mol/kg-soln if necessary.

## ABOUT THE PROGRAMS

There are four programs in this package:

CO2SYSTM.EXE
FCO2TCO2 .EXE
PHTCO2 .EXE
CO2BTCH.EXE.

They are designed to be run on any 80x86 computer equipped with the DOS operating system. They are run simply by typing the program name from the command line. All input data for these programs should be in micromoles per kilogram of solution ($\mu$mol/kg-soln), micro-atmospheres ($\mu$atm) in the case of $fCO_2$ , and parts per million (ppm) in the case of $xCO_2$. The pH should be on the total scale. All outputs are in these units also. Each program in this package allows a choice from the four sets of constants $K_1$ and $K_2$: those of Roy et al. (1993), Goyet and Poisson (1989), Hansson (1973, as refit by Dickson and Millero, 1987), and Mehrbach et al. (1973, as refit by Dickson and Millero, 1987).

The program CO2SYSTM.EXE allows the calculation of any pair of $CO_2$ system variables (TA, TC, pH, and $fCO_2$ or $xCO_2$ ) from the other two. It requires input of the two known carbonate system parameters, temperature, salinity, and the total concentrations of both phosphate and silicate. The other two parameters of the $CO_2$ system are then calculated, as well as partials with respect to the two input parameters, temperature, salinity, and the constants $K_0$, $K_1$, and $K_2$. For example, the result for $fCO_2$ may appear as:

$$fCO_2 = 446.3 + 0.2\ dTC + -1.1\ dmpH + 1.0\ dTemp + -1.3\ dSAL + -4.4\ \%dK_0 + -4.4\ \%dK_1 + -0.4\ \%dK_2$$

This means that for the input values of TC (2000) and pH (8.0), $fCO_2$ has a value of 446.3 $\mu$atm

(micro atmospheres), and increases by .2 $\mu$atm for each 1 $\mu$mol/kg-soln increase of TC, decreases by 1.1 $\mu$atm for each .001 unit increase in pH (mpH refers to milli-pH), increases 1.0 $\mu$atm for each 1 deg change in temperature, and decreases by 1.3 $\mu$atm for a unit increase in salinity in dimensionless salinity units (often referred to as parts per thousand). The next line means that for each one percent change in $K_0$ (shown as $\%dK_0$), $fCO_2$ decreases by 4.4 uatm, and similarly for each one percent change in $K_1$ and $K_2$ ($\%dK_1$ and $\%dK_2$). The program lists the estimated ACCURACY of $K_0$ and the 2s (two standard deviation) PRECISION of the constants $K_1$ and $K_2$ to allow an estimate to be made of the uncertainty of the final answer ($fCO_2$ in this case) due to the uncertainty in the equilibrium constants.

The program also gives the contribution to the total alkalinity from the phosphate, silicate, boron, and carbonate species. There is a slight contribution to the alkalinity from hydroxide (which is not listed), so the sum of the alkalinities listed may not equal the total alkalinity given.

The two programs FCO2TCO2.EXE and PHTCO2.EXE are similar in operation, but the former is for $fCO_2$ (or $xCO_2$) and the latter is for pH. Both require knowledge of TC, which is the carbonate system parameter most likely to be known. Each allows a choice of carbonate constants and of $fCO_2$ or $xCO_2$ as the desired variable. Recall $fCO_2$ is referenced to wet air and $xCO_2$ is referenced to dry air. The values of salinity, TC, $fCO_2$ or $xCO_2$ (or pH), and two temperatures, the temperature of measurement and a reference temperature (which can be any other temperature at which the values are desired) are input from the screen. The programs find the values TA, $fCO_2$ or $xCO_2$ (or pH) at this reference temperature, and pH (or $fCO_2$ or $xCO_2$) at both the temperature of measurement and at the reference temperature, as well as the partials with respect to salinity, TC, the input $fCO_2$ or $xCO_2$ (or pH), the measured temperature, and the constants $K_1$ and $K_2$ (also $K_0$ for FCO2TCO2.EXE). The partials with respect to the constants are evaluated at the temperature of measurement only. Care and thought should therefore go into the interpretation and use of these partials.

The program CO2BTCH.EXE is similar to CO2SYSTM.EXE, but allows data to be batch input from ASCII files. This would be useful in the case of large data sets. This program will perform the same calculations as the programs FCO2TCO2.EXE and PHTCO2.EXE, however it does not calculate and report partials. It allows an interactive choice of which pair of $CO_2$ system parameters should be input, as well as which carbonate constants should be used. The input filename and the filename to which the data should be written are input interactively. This program was designed for easy use with data output as text files from Microsoft EXCEL or other spreadsheets.

The input data file must contain two header lines. Each row contains the input data for one sample. This data may be space separated or comma separated. The first entry must be an ID string of up to six letters. If the data are space separated, this MUST be in double quotes. If the data are comma separated this need not be so. The ID string is followed by two temperatures, the first the temperature of measurement, and the second a reference at which pH and $fCO_2$ or $xCO_2$ are to be calculated. The two temperatures should be followed by the

salinity, the two known $CO_2$ system parameters, the total phosphate, and the total silicate. These last two are often not quantitatively important, but a value must be entered in these columns. Again, all paramters must be provided with units of $\mu$mol/kg-soln, $\mu$atm or ppm, and with pH on the total scale. The order in which the parameters are to be provided is TA before TC before pH before $fCO_2$ (or $xCO_2$). This means that for whichever two $CO_2$ system parameters are provided, TA will always come first if it is used, TC (if it is used) will precede any other parameter except TA, etc. Care must be taken to insure that the input data is in the correct format. Six example data files: CASE1.INP, CASE2.INP, CASE3.INP, CASE4.INP, CASE5.INP, and CASE6.INP are included on the disk as examples for each of the cases of $CO_2$ system parameter pairs. CASE1.INP works for cases 1 and 7, CASE2.INP works for cases 2 and 8, etc. Cases 7 through 12 are the same as cases 1 through 6 except they use $xCO_2$ in dry air instead of $fCO_2$ in wet (H2O-saturated) air.

The output data are space separated. There are two header lines in the output file. One has the name of the input file and the set of carbonate constants which were used in the calculation, and the other has headings for the columns. The data format is one line per sample. The first entry is the ID number, followed by the temperature of measurement, the salinity, the two $CO_2$ system paramerers, the total phosphate, the total silicate, the other two (calculated) carbonate system parameters, the reference temperature, and the pH and $fCO_2$ or $xCO_2$ evaluated at that reference temperature.

## FURTHER INFORMATION

This set of programs was written by
Ernie Lewis
Building 318
Brookhaven National Laboratory
Upton, New York 11973
516-282-7706
ELewis@bnl.gov

Every effort was made to insure that the programs are error-free and as user-friendly as possible. Please report any errors or problems to the above address. For more information on the $CO_2$ system in seawater, refer to:

DOE (1994) Handbook of methods for the analysis of the various parameters of the carbon dioxide system in sea water; version 2, A. G. Dickson and C. Goyet, eds. ORNL/CDIAC-74,

on which this program was based, or

Skirrow, G. (1975) The dissolved gases -- carbon dioxide. In: Chemical Oceanography, Vol. 2, J. P. Riley and G. Skirrow, eds., Academic Press, London, pp. 1 - 192.

## REFERENCES

The following references were cited:


Dickson, A.G. and Millero, F.W., A comparison of the equilibrium constants for the dissociation of carbonic acid in seawater media, Deep-Sea Research, 34:1733-1743, 1987.

Dickson, A.G. and Riley, J.P., The effect of analytical error on the evaluation of the components of the aquatic carbon-dioxide system, Marine Chemistry, 6:77-85, 1978.

Goyet, C. and Poisson, A., New determination of carbonic acid dissociation constants in seawater as a function of temperature and salinity, Deep-Sea Research 36:1635-1654, 1989.

Hansson, I., A new set of acidity constants for carbonic acid and boric acid in seawater, Deep-Sea Research 20:461-478, 1973.

Mehrbach, C. et al., Measurement of the apparent dissociation constants of carbonic acid in seawater at atmospheric pressure, Limnology and Oceanography, 18:897-907, 1973.

Murray, C.N. and Riley, J.P., The solubility of gases in distilled water and seawater - IV. Carbon dioxide, Deep-Sea Research, 18:533-541, 1971.

Park, P. K., Oceanic CO2 system: an evaluation of ten methods of investigation, Limnology and Oceanography, 14:179-186, 1969.

Roy, R. N., et al., The dissociation constants of carbonic acid in seawater at salinities 5 to 45 and temperatures 0 to 45 C, Marine Chemistry 44:249-267, 1993.

Weiss, R. F., Carbon dioxide in water and seawater: the solubility of a non-ideal gas, Marine Chemistry, 2:203-215, 1974.

Weiss, R. F., and Price, B. A., Nitrous oxide solubility in water and seawater, Marine Chemistry 8, 347-359, 1980.


## ACKNOWLEDGEMENTS


Significant help, advice, and clarification was supplied by Dr. Andrew Dickson of Scripps Institution of Oceanography, and Kenneth Johnson of Brookhaven National Laboratory, during the assembly of these programs. This work was supported by the US Department of Energy under contract DE-ACO2-76CH00016, through a project entitled "Inorganic Carbon for the World Ocean Circulation Experiment - World Hydrographic Program" (D.W.R. Wallace and K.M. Johnson, PIs).

APPENDIX

# Source Code Listing for CO2SYSTM.BAS

# Version 1.02

(Subroutines used for $CO_2$ system calculations are common among all of the programs included in this package.)

```
' PROGRAM CO2SYSTM.BAS, version 1.02, 5-9-95, written by ERNIE LEWIS
'
'
' THIS IS A QBASIC PROGRAM TO FIND ANY TWO PARAMETERS OF THE CO2 SYSTEM
'      (TA, TC, pH, AND fCO2 OR xCO2) FROM THE OTHER TWO.
' PARTIALS ARE INCLUDED TO SHOW SENSITIVITY TO INPUTS AND K0, K1, AND K2.
' THE PRESSURE IS ASSUMED TO BE 1 ATM.
' FOR MORE INFORMATION SEE THE SUB AboutThisProgram OR THE PROGRAM
'    CO2SYSTM.TXT.
'
'
'******************
DeclareStuff:
    DECLARE SUB AboutThisProgram ()
    DECLARE SUB ChooseWhichCase (ICase%)
    DECLARE SUB ChooseWhichKs (WhichKs%, WhoseK$, KComment1$, KComment2$)
    DECLARE SUB Constants (TempC, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
    DECLARE SUB ErrorSub (ERROR$)
    DECLARE SUB FindAlkContributions (pH, TC, K(), T(), BAlk, CAlk, PhosAlk, SiAlk)
    DECLARE SUB FindfCO2fromTCpH (TC, pH, K0, K1, K2, fCO2)
    DECLARE SUB FindpHfromTAfCO2 (TA, fCO2, K0, K(), T(), Z, pH)
    DECLARE SUB FindpHfromTATC (TA, TC, K(), T(), Z, pH)
    DECLARE SUB FindpHfromTCfCO2 (TC, fCO2, K0, K1, K2, pH)
    DECLARE SUB FindTAfromTCpH (TC, pH, K(), T(), Z, TA)
    DECLARE SUB FindTCfromTApH (TA, pH, K(), T(), Z, TC)
    DECLARE SUB InputStuff (ICase%, TempC, SAL, PRES, TP, TSi)
    DECLARE SUB SetParameters (dTA, dTC, dpH, dfCO2, dTempC, dSAL, pcdK0, pcdK1, pcdK2, K0Factor,
K1Factor, K2Factor)
'
'
'******************
DimensionStuff:
    DIM K(10):          ' THESE ARE THE EQUILIBRIUM CONSTANTS
    DIM T(5):           ' THESE ARE THE AMOUNTS OF STUFF
'
'
'******************
    CALL AboutThisProgram
    CALL SetParameters(dTA, dTC, dpH, dfCO2, dTempC, dSAL, pcdK0, pcdK1, pcdK2, K0Factor, K1Factor,
K2Factor)
Start:
    ON ERROR GOTO ErrorHandler:
    CALL ChooseWhichCase(ICase%)
    CALL ChooseWhichKs(WhichKs%, WhoseK$, KComment1$, KComment2$)
    CALL InputStuff(ICase%, TempC, SAL, PRES, TP, TSi)
    T(4) = TP * 10 ^ (-6)
    T(5) = TSi * 10 ^ (-6)
    CALL Constants(TempC, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
    K1 = K(1): K2 = K(2)
'
'
    SELECT CASE ICase%
'   ************************************************************************
```

```
CASE 1, 7
    INPUT "  Enter TA (umol/kg-soln): ", TA$: IF LEN(TA$) <> 0 THEN TA = VAL(TA$) * 10 ^ (-6)
    LOCATE 8, 1: PRINT USING "  TA  = ####.# "; TA * 10 ^ 6; : PRINT SPACE$(20)
    INPUT "  Enter TC (umol/kg-soln): ", TC$: IF LEN(TC$) <> 0 THEN TC = VAL(TC$) * 10 ^ (-6)
    LOCATE 9, 1: PRINT USING "  TC  = ####.# "; TC * 10 ^ 6; : PRINT SPACE$(20)
'
'   DO AT GIVEN TA, TC
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH): pH0 = pH
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2)
        xCO2 = fCO2 / (FugFac * VPFac)
        fCO20 = fCO2: xCO20 = xCO2
'   INCREASE TA BY dTA
        TAnew = TA + dTA
        CALL FindpHfromTATC(TAnew, TC, K0, T0, Z, pH): dpHdTA = (pH - pH0) / dTA
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2dTA = (fCO2 - fCO20) / dTA: dxCO2dTA = (xCO2 - xCO20) / dTA
'   INCREASE TC BY dTC
        TCnew = TC + dTC
        CALL FindpHfromTATC(TA, TCnew, K0, T0, Z, pH): dpHdTC = (pH - pH0) / dTC
        CALL FindfCO2fromTCpH(TCnew, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2dTC = (fCO2 - fCO20) / dTC: dxCO2dTC = (xCO2 - xCO20) / dTC
'   INCREASE K0 BY pcdK0 %
        K0new = K0 * K0Factor
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH)
        CALL FindfCO2fromTCpH(TC, pH, K0new, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2pcdK0 = (fCO2 - fCO20) / pcdK0: dxCO2pcdK0 = (xCO2 - xCO20) / pcdK0
'   INCREASE K1 BY pcdK1 %
        K(1) = K(1) * K1Factor: K1 = K(1)
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH): dpHpcdK1 = (pH - pH0) / pcdK1
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2pcdK1 = (fCO2 - fCO20) / pcdK1: dxCO2pcdK1 = (xCO2 - xCO20) / pcdK1
        K(1) = K(1) / K1Factor: K1 = K(1)
'   INCREASE K2 BY pcdK2 %
        K(2) = K(2) * K2Factor: K2 = K(2)
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH): dpHpcdK2 = (pH - pH0) / pcdK2
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2pcdK2 = (fCO2 - fCO20) / pcdK2: dxCO2pcdK2 = (xCO2 - xCO20) / pcdK2
        K(2) = K(2) / K2Factor: K2 = K(2)
'   INCREASE TempC BY dTempC
        TempCnew = TempC + dTempC
        CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K0, T0, Z, NERNST, FugFac, VPFac)
        K1 = K(1): K2 = K(2)
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH): dpHdTempC = (pH - pH0) / dTempC
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2dTempC = (fCO2 - fCO20) / dTempC: dxCO2dTempC = (xCO2 - xCO20) / dTempC
'   INCREASE SAL BY dSAL
        SALnew = SAL + dSAL
        CALL Constants(TempC, SALnew, PRES, D, K0, WhichKs%, K0, T0, Z, NERNST, FugFac, VPFac)
        K1 = K(1): K2 = K(2)
        CALL FindpHfromTATC(TA, TC, K0, T0, Z, pH): dpHdSAL = (pH - pH0) / dSAL
        CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
        dfCO2dSAL = (fCO2 - fCO20) / dSAL: dxCO2dSAL = (xCO2 - xCO20) / dSAL
'
```

```
        LOCATE 11, 1
        PRINT USING "   pH  = #.#### + #.#### dTA + #.#### dTC  + #.#### dTemp + #.#### dSAL "; pH0;
dpHdTA * 10 ^ (-6); dpHdTC * 10 ^ (-6); dpHdTempC; dpHdSAL
        PRINT USING "                                  + #.#### %dK1  + #.#### %dK2 "; dpHpcdK1; dpHpcdK2
        PRINT
        IF ICase% = 1 THEN
            PRINT USING "   fCO2 = ####.# + ###.# dTA +  ###.# dTC  +  ###.# dTemp + ###.# dSAL ";
fCO20 * 10 ^ 6; dfCO2dTA; dfCO2dTC; dfCO2dTempC * 10 ^ 6; dfCO2dSAL * 10 ^ 6
            PRINT USING "                          + ###.# %dK0 + ###.# %dK1  + ###.# %dK2 "; dfCO2pcdK0 *
10 ^ 6; dfCO2pcdK1 * 10 ^ 6; dfCO2pcdK2 * 10 ^ 6
        END IF
        IF ICase% = 7 THEN
            PRINT USING "   xCO2 = ####.# + ###.# dTA +  ###.# dTC  +  ###.# dTemp + ###.# dSAL ";
xCO20 * 10 ^ 6; dxCO2dTA; dxCO2dTC; dxCO2dTempC * 10 ^ 6; dxCO2dSAL * 10 ^ 6
            PRINT USING "                          + ###.# %dK0 + ###.# %dK1  + ###.# %dK2 "; dxCO2pcdK0 *
10 ^ 6; dxCO2pcdK1 * 10 ^ 6; dxCO2pcdK2 * 10 ^ 6
        END IF
        pH = pH0: fCO2 = fCO20: xCO2 = xCO20
'
'
'   ************************************************************************
    CASE 2, 8
        INPUT "   Enter TA (umol/kg-soln): ", TA$: IF LEN(TA$) <> 0 THEN TA = VAL(TA$) * 10 ^ (-6)
        LOCATE 8, 1: PRINT USING "   TA  = ####.# "; TA * 10 ^ 6; : PRINT SPACE$(20)
        INPUT "   Enter pH: ", pH$: IF LEN(pH$) <> 0 THEN pH = VAL(pH$)
        LOCATE 9, 1: PRINT USING "   pH  = #.####     "; pH
'
'       DO AT GIVEN TA, pH
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC): TC0 = TC
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2)
            xCO2 = fCO2 / (FugFac * VPFac)
            fCO20 = fCO2: xCO20 = xCO2
'       INCREASE TA BY dTA
            TAnew = TA + dTA
            CALL FindTCfromTApH(TAnew, pH, K0, T0, Z, TC): dTCdTA = (TC - TC0) / dTA
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dTA = (fCO2 - fCO20) / dTA: dxCO2dTA = (xCO2 - xCO20) / dTA
'       INCREASE pH BY dpH
            pHnew = pH + dpH
            CALL FindTCfromTApH(TA, pHnew, K0, T0, Z, TC): dTCdpH = (TC - TC0) / dpH
            CALL FindfCO2fromTCpH(TC, pHnew, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dpH = (fCO2 - fCO20) / dpH: dxCO2dpH = (xCO2 - xCO20) / dpH
'       INCREASE K0 BY pcdK0 %
            K0new = K0 * K0Factor
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC)
            CALL FindfCO2fromTCpH(TC, pH, K0new, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2pcdK0 = (fCO2 - fCO20) / pcdK0: dxCO2pcdK0 = (xCO2 - xCO20) / pcdK0
'       INCREASE K1 BY pcdK1 %
            K(1) = K(1) * K1Factor: K1 = K(1)
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC): dTCpcdK1 = (TC - TC0) / pcdK1
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2pcdK1 = (fCO2 - fCO20) / pcdK1: dxCO2pcdK1 = (xCO2 - xCO20) / pcdK1
            K(1) = K(1) / K1Factor: K1 = K(1)
```

```
'       INCREASE K2 BY pcdK2 %
            K(2) = K(2) * K2Factor: K2 = K(2)
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC): dTCpcdK2 = (TC - TC0) / pcdK2
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2pcdK2 = (fCO2 - fCO20) / pcdK2: dxCO2pcdK2 = (xCO2 - xCO20) / pcdK2
            K(2) = K(2) / K2Factor: K2 = K(2)
'       INCREASE TempC BY dTempC
            TempCnew = TempC + dTempC
            CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K0, T0, Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC): dTCdTempC = (TC - TC0) / dTempC
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dTempC = (fCO2 - fCO20) / dTempC: dxCO2dTempC = (xCO2 - xCO20) / dTempC
'       INCREASE SAL BY dSAL
            SALnew = SAL + dSAL
            CALL Constants(TempC, SALnew, PRES, D, K0, WhichKs%, K0, T0, Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindTCfromTApH(TA, pH, K0, T0, Z, TC): dTCdSAL = (TC - TC0) / dSAL
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dSAL = (fCO2 - fCO20) / dSAL: dxCO2dSAL = (xCO2 - xCO20) / dSAL
'
        LOCATE 11, 1
        PRINT USING "   TC  = ####.# + ###.# dTA + ###.# dmpH +  ###.# dTemp +  ###.# dSAL "; TC0 * 10 ^
6; dTCdTA; dTCdpH * 10 ^ 6 / 10 ^ 3; dTCdTempC * 10 ^ 6; dTCdSAL * 10 ^ 6
        PRINT USING "                                + ###.# %dK1  + ###.# %dK2 "; dTCpcdK1 * 10 ^ 6;
dTCpcdK2 * 10 ^ 6
        PRINT
        IF ICase% = 2 THEN
            PRINT USING "   fCO2 = ####.# + ###.# dTA + ###.# dmpH +  ###.# dTemp +  ###.# dSAL ";
fCO20 * 10 ^ 6; dfCO2dTA; dfCO2dpH * 10 ^ 6 / 10 ^ 3; dfCO2dTempC * 10 ^ 6; dfCO2dSAL * 10 ^ 6
            PRINT USING "                          + ###.# %dK0 +  ###.# %dK1  + ###.# %dK2 "; dfCO2pcdK0 * 10
^ 6; dfCO2pcdK1 * 10 ^ 6; dfCO2pcdK2 * 10 ^ 6
        END IF
        IF ICase% = 8 THEN
            PRINT USING "   xCO2 = ####.# + ###.# dTA + ###.# dmpH +  ###.# dTemp +  ###.# dSAL ";
xCO20 * 10 ^ 6; dxCO2dTA; dxCO2dpH * 10 ^ 6 / 10 ^ 3; dxCO2dTempC * 10 ^ 6; dxCO2dSAL * 10 ^ 6
            PRINT USING "                          + ###.# %dK0 +  ###.# %dK1  + ###.# %dK2 "; dxCO2pcdK0 * 10
^ 6; dxCO2pcdK1 * 10 ^ 6; dxCO2pcdK2 * 10 ^ 6
        END IF
        TC = TC0: fCO2 = fCO20: xC02 = xCO20
'
'
'   ************************************************************************
    CASE 3, 9
        INPUT "   Enter TA (umol/kg-soln): ", TA$: IF LEN(TA$) <> 0 THEN TA = VAL(TA$) * 10 ^ (-6)
        LOCATE 8, 1: PRINT USING "   TA  = ####.# "; TA * 10 ^ 6; : PRINT SPACE$(20)
        IF ICase% = 3 THEN
            INPUT "   Enter fCO2 (u-atm): ", fCO2$: IF LEN(fCO2$) <> 0 THEN fCO2 = VAL(fCO2$) * 10 ^
(-6)
            LOCATE 9, 1: PRINT USING "   fCO2 = ####.# "; fCO2 * 10 ^ 6; : PRINT SPACE$(20)
            xCO2 = fCO2 / (FugFac * VPFac)
        END IF
        IF ICase% = 9 THEN
```

```
        INPUT "   Enter xCO2 (ppm): ", xCO2$: IF LEN(xCO2$) <> 0 THEN xCO2 = VAL(xCO2$) * 10 ^
(-6)
        LOCATE 9, 1: PRINT USING "   xCO2 = ####.# "; xCO2 * 10 ^ 6; : PRINT SPACE$(20)
        fCO2 = xCO2 * FugFac * VPFac
    END IF
'
'   DO AT GIVEN TA, fCO2 OR xCO2
        CALL FindpHfromTAfCO2(TA, fCO2, K0, K(), T(), Z, pH): pH0 = pH
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): TC0 = TC
'       OR: H = 10^(-pH):TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): TC0 = TC
'   INCREASE TA BY dTA
        TAnew = TA + dTA
        CALL FindpHfromTAfCO2(TAnew, fCO2, K0, K(), T(), Z, pH): dpHdTA = (pH - pH0) / dTA
        CALL FindTCfromTApH(TAnew, pH, K(), T(), Z, TC): dTCdTA = (TC - TC0) / dTA
'   INCREASE fCO2 BY dfCO2
        fCO2new = fCO2 + dfCO2: dxCO2 = dfCO2 / (FugFac * VPFac)
        CALL FindpHfromTAfCO2(TA, fCO2new, K0, K(), T(), Z, pH)
        dpHdfCO2 = (pH - pH0) / dfCO2: dpHdxCO2 = (pH - pH0) / dxCO2
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC)
        dTCdfCO2 = (TC - TC0) / dfCO2: dTCdxCO2 = (TC - TC0) / dxCO2
'   INCREASE K0 BY pcdK0 %
        K0new = K0 * K0Factor
        CALL FindpHfromTAfCO2(TA, fCO2, K0new, K(), T(), Z, pH): dpHpcdK0 = (pH - pH0) / pcdK0
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): dTCpcdK0 = (TC - TC0) / pcdK0
'   INCREASE K1 BY pcdK1 %
        K(1) = K(1) * K1Factor: K1 = K(1)
        CALL FindpHfromTAfCO2(TA, fCO2, K0, K(), T(), Z, pH): dpHpcdK1 = (pH - pH0) / pcdK1
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): dTCpcdK1 = (TC - TC0) / pcdK1
        K(1) = K(1) / K1Factor: K1 = K(1)
'   INCREASE K2 BY pcdK2 %
        K(2) = K(2) * K2Factor: K2 = K(2)
        CALL FindpHfromTAfCO2(TA, fCO2, K0, K(), T(), Z, pH): dpHpcdK2 = (pH - pH0) / pcdK2
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): dTCpcdK2 = (TC - TC0) / pcdK2
        K(2) = K(2) / K2Factor: K2 = K(2)
'   INCREASE TempC BY dTempC
        TempCnew = TempC + dTempC
        CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
        K1 = K(1): K2 = K(2)
        CALL FindpHfromTAfCO2(TA, fCO2, K0, K(), T(), Z, pH): dpHdTempC = (pH - pH0) / dTempC
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): dTCdTempC = (TC - TC0) / dTempC
'   INCREASE SAL BY dSAL
        SALnew = SAL + dSAL
        CALL Constants(TempC, SALnew, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
        K1 = K(1): K2 = K(2)
        CALL FindpHfromTAfCO2(TA, fCO2, K0, K(), T(), Z, pH): dpHdSAL = (pH - pH0) / dSAL
        CALL FindTCfromTApH(TA, pH, K(), T(), Z, TC): dTCdSAL = (TC - TC0) / dSAL
'
    LOCATE 11, 1
    IF ICase% = 3 THEN
        PRINT USING "   TC  = ####.# +  ###.# dTA +  ###.# dfCO2 +  ###.# dTemp +  ###.# dSAL ";
TC0 * 10 ^ 6; dTCdTA; dTCdfCO2; dTCdTempC * 10 ^ 6; dTCdSAL * 10 ^ 6
        PRINT USING "                    +  ###.# %dK0 +  ###.# %dK1 +  ###.# %dK2 "; dTCpcdK0 *
10 ^ 6; dTCpcdK1 * 10 ^ 6; dTCpcdK2 * 10 ^ 6
```

```
            PRINT
            PRINT USING "   pH  = #.#### + #.#### dTA + #.#### dfCO2 + #.#### dTemp + #.#### dSAL ";
pH0; 10 ^ (-6) * dpHdTA; 10 ^ (-6) * dpHdfCO2; dpHdTempC; dpHdSAL
            PRINT USING "                    + #.#### %dK0 + #.#### %dK1 + #.#### %dK2 "; dpHpcdK0;
dpHpcdK1; dpHpcdK2
        END IF
        IF ICase% = 9 THEN
            PRINT USING "   TC  = ####.# +  ###.# dTA +  ###.# dxCO2 +  ###.# dTemp +  ###.# dSAL ";
TC0 * 10 ^ 6; dTCdTA; dTCdxCO2; dTCdTempC * 10 ^ 6; dTCdSAL * 10 ^ 6
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTCpcdK0 *
10 ^ 6; dTCpcdK1 * 10 ^ 6; dTCpcdK2 * 10 ^ 6
            PRINT
            PRINT USING "   pH  = #.#### + #.#### dTA + #.#### dxCO2 + #.#### dTemp + #.#### dSAL ";
pH0; 10 ^ (-6) * dpHdTA; 10 ^ (-6) * dpHdxCO2; dpHdTempC; dpHdSAL
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTCpcdK0 *
10 ^ 6; dTCpcdK1 * 10 ^ 6; dTCpcdK2 * 10 ^ 6
        END IF
        TC = TC0: pH = pH0
'
'
'   ************************************************************************
    CASE 4, 10
        INPUT "   Enter TC (umol/kg-soln): ", TC$: IF LEN(TC$) <> 0 THEN TC = VAL(TC$) * 10 ^ (-6)
        LOCATE 8, 1: PRINT USING "   TC  = ####.# "; TC * 10 ^ 6; : PRINT SPACE$(20)
        INPUT "   Enter pH: ", pH$: IF LEN(pH$) <> 0 THEN pH = VAL(pH$)
        LOCATE 9, 1: PRINT USING "   pH  = #.####     "; pH
'
'       DO AT GIVEN TC, pH
            CALL FindTAfromTCpH(TC, pH, K0, T0, Z, TA): TA0 = TA
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2)
            xCO2 = fCO2 / (FugFac * VPFac)
            fCO20 = fCO2: xCO20 = xCO2
'       INCREASE TC BY dTC
            TCnew = TC + dTC
            CALL FindTAfromTCpH(TCnew, pH, K0, T0, Z, TA): dTAdTC = (TA - TA0) / dTC
            CALL FindfCO2fromTCpH(TCnew, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dTC = (fCO2 - fCO20) / dTC: dxCO2dTC = (xCO2 - xCO20) / dTC
'       INCREASE pH BY dpH
            pHnew = pH + dpH
            CALL FindTAfromTCpH(TC, pHnew, K0, T0, Z, TA): dTAdpH = (TA - TA0) / dpH
            CALL FindfCO2fromTCpH(TC, pHnew, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dpH = (fCO2 - fCO20) / dpH: dxCO2dpH = (xCO2 - xCO20) / dpH
'       INCREASE K0 BY pcdK0 %
            K0new = K0 * K0Factor
            CALL FindTAfromTCpH(TC, pH, K0, T0, Z, TA)
            CALL FindfCO2fromTCpH(TC, pH, K0new, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2pcdK0 = (fCO2 - fCO20) / pcdK0: dxCO2pcdK0 = (xCO2 - xCO20) / pcdK0
'       INCREASE K1 BY pcdK1 %
            K(1) = K(1) * K1Factor: K1 = K(1)
            CALL FindTAfromTCpH(TC, pH, K0, T0, Z, TA): dTApcdK1 = (TA - TA0) / pcdK1
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2pcdK1 = (fCO2 - fCO20) / pcdK1: dxCO2pcdK1 = (xCO2 - xCO20) / pcdK1
            K(1) = K(1) / K1Factor: K1 = K(1)
```

```
'       INCREASE K2 BY pcdK2 %
            K(2) = K(2) * K2Factor: K2 = K(2)
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK2 = (TA - TA0) / pcdK2
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
             dfCO2pcdK2 = (fCO2 - fCO20) / pcdK2: dxCO2pcdK2 = (xCO2 - xCO20) / pcdK2
            K(2) = K(2) / K2Factor: K2 = K(2)
'       INCREASE TempC BY dTempC
            TempCnew = TempC + dTempC
            CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdTempC = (TA - TA0) / dTempC
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dTempC = (fCO2 - fCO20) / dTempC: dxCO2dTempC = (xCO2 - xCO20) / dTempC
'       INCREASE SAL BY dSAL
            SALnew = SAL + dSAL
            CALL Constants(TempC, SALnew, PRES; D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdSAL = (TA - TA0) / dSAL
            CALL FindfCO2fromTCpH(TC, pH, K0, K1, K2, fCO2): xCO2 = fCO2 / (FugFac * VPFac)
            dfCO2dSAL = (fCO2 - fCO20) / dSAL: dxCO2dSAL = (xCO2 - xCO20) / dSAL
'
        LOCATE 11, 1
        PRINT USING "   TA  = ####.# + ###.# dTC + ###.# dmpH + ###.# dTemp + ###.# dSAL "; TA0 * 10
^ 6; dTAdTC; dTAdpH * 10 ^ 3; dTAdTempC * 10 ^ 6; dTAdSAL * 10 ^ 6
        PRINT USING "                                + ###.# %dK1  + ###.# %dK2 "; dTApcdK1 * 10 ^ 6;
dTApcdK2 * 10 ^ 6
        PRINT
        IF ICase% = 4 THEN
            PRINT USING "   fCO2 = ####.# + ###.# dTC + ###.# dmpH +  ##.# dTemp +  ##.# dSAL ";
fCO20 * 10 ^ 6; dfCO2dTC; dfCO2dpH * 10 ^ 3; dfCO2dTempC * 10 ^ 6; dfCO2dSAL * 10 ^ 6
            PRINT USING "                       + ###.# %dK0 + ###.# %dK1  + ###.# %dK2 "; dfCO2pcdK0 *
10 ^ 6; dfCO2pcdK1 * 10 ^ 6; dfCO2pcdK2 * 10 ^ 6
        END IF
        IF ICase% = 10 THEN
            PRINT USING "   xCO2 = ####.# + ###.# dTC + ###.# dmpH +  ##.# dTemp +  ##.# dSAL ";
xCO20 * 10 ^ 6; dxCO2dTC; dxCO2dpH * 10 ^ 3; dxCO2dTempC * 10 ^ 6; dxCO2dSAL * 10 ^ 6
            PRINT USING "                       + ###.# %dK0 + ###.# %dK1  + ###.# %dK2 "; dxCO2pcdK0 *
10 ^ 6; dxCO2pcdK1 * 10 ^ 6; dxCO2pcdK2 * 10 ^ 6
        END IF
        TA = TA0: fCO2 = fCO20: xCO2 = xCO20
'
'
'   ***********************************************************************
    CASE 5, 11
        INPUT "   Enter TC (umol/kg-soln): ", TC$: IF LEN(TC$) <> 0 THEN TC = VAL(TC$) * 10 ^ (-6)
        LOCATE 8, 1: PRINT USING "   TC  = ####.# "; TC * 10 ^ 6; : PRINT SPACE$(20)
        IF ICase% = 5 THEN
            INPUT "   Enter fCO2 (u-atm): ", fCO2$: IF LEN(fCO2$) <> 0 THEN fCO2 = VAL(fCO2$) * 10 ^
(-6)
            LOCATE 9, 1: PRINT USING "   fCO2 = ####.# "; fCO2 * 10 ^ 6; : PRINT SPACE$(20)
            xCO2 = fCO2 / (FugFac * VPFac)
        END IF
        IF ICase% = 11 THEN
```

```
            INPUT "   Enter xCO2 (ppm): ", xCO2$: IF LEN(xCO2$) <> 0 THEN xCO2 = VAL(xCO2$) * 10 ^
(-6)
            LOCATE 9, 1: PRINT USING "   xCO2 = ####.# "; xCO2 * 10 ^ 6; : PRINT SPACE$(20)
            fCO2 = xCO2 * FugFac * VPFac
        END IF
'
'       DO AT GIVEN TC, fCO2 OR xCO2
            CALL FindpHfromTCfCO2(TC, fCO2, K0, K1, K2, pH): pH0 = pH
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): TA0 = TA
'       INCREASE TC BY dTC
            TCnew = TC + dTC
            CALL FindpHfromTCfCO2(TCnew, fCO2, K0, K1, K2, pH): dpHdTC = (pH - pH0) / dTC
            CALL FindTAfromTCpH(TCnew, pH, K(), T(), Z, TA): dTAdTC = (TA - TA0) / dTC
'       INCREASE fCO2 BY dfCO2
            fCO2new = fCO2 + dfCO2: dxCO2 = dfCO2 / (FugFac * VPFac)
            CALL FindpHfromTCfCO2(TC, fCO2new, K0, K1, K2, pH)
            dpHdfCO2 = (pH - pH0) / dfCO2: dpHdxCO2 = (pH - pH0) / dxCO2
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA)
            dTAdfCO2 = (TA - TA0) / dfCO2: dTAdxCO2 = (TA - TA0) / dxCO2
'       INCREASE K0 BY pcdK0 %
            K0new = K0 * K0Factor
            CALL FindpHfromTCfCO2(TC, fCO2, K0new, K1, K2, pH): dpHpcdK0 = (pH - pH0) / pcdK0
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK0 = (TA - TA0) / pcdK0
'       INCREASE K1 BY pcdK1 %
            K(1) = K(1) * K1Factor: K1 = K(1)
            CALL FindpHfromTCfCO2(TC, fCO2, K0, K1, K2, pH): dpHpcdK1 = (pH - pH0) / pcdK1
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK1 = (TA - TA0) / pcdK1
            K(1) = K(1) / K1Factor: K1 = K(1)
'       INCREASE K2 BY pcdK2 %
            K(2) = K(2) * K2Factor: K2 = K(2)
            CALL FindpHfromTCfCO2(TC, fCO2, K0, K1, K2, pH): dpHpcdK2 = (pH - pH0) / pcdK2
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK2 = (TA - TA0) / pcdK2
            K(2) = K(2) / K2Factor: K2 = K(2)
'       INCREASE TempC BY dTempC
            TempCnew = TempC + dTempC
            CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindpHfromTCfCO2(TC, fCO2, K0, K1, K2, pH): dpHdTempC = (pH - pH0) / dTempC
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdTempC = (TA - TA0) / dTempC
'       INCREASE SAL BY dSAL
            SALnew = SAL + dSAL
            CALL Constants(TempC, SALnew, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            CALL FindpHfromTCfCO2(TC, fCO2, K0, K1, K2, pH): dpHdSAL = (pH - pH0) / dSAL
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdSAL = (TA - TA0) / dSAL
'
        LOCATE 11, 1
        IF ICase% = 5 THEN
            PRINT USING "   TA = ####.# +  ###.# dTC +  ###.# dfCO2 +  ###.# dTemp + ###.# dSAL ";
TA0 * 10 ^ 6; dTAdTC; dTAdfCO2; dTAdTempC * 10 ^ 6; dTAdSAL * 10 ^ 6
            PRINT USING "                    +  ###.# %dK0 +  ###.# %dK1 +  ###.# %dK2 "; dTApcdK0 *
10 ^ 6; dTApcdK1 * 10 ^ 6; dTApcdK2 * 10 ^ 6
            PRINT
```

```
            PRINT USING "   pH  =  #.####  +  #.####  dTC  +  #.####  dfCO2  +  #.####  dTemp  +  #.####  dSAL ";
pH0; dpHdTC * 10 ^ (-6); dpHdfCO2 * 10 ^ (-6); dpHdTempC; dpHdSAL
            PRINT USING "                          +  #.####  %dK0  +  #.####  %dK1  +  #.####  %dK2 "; dpHpcdK0;
dpHpcdK1; dpHpcdK2
        END IF
        IF ICase% = 11 THEN
            PRINT USING "   TA  =  ####.#  +  ###.#  dTC  +  ###.#  dxCO2  +  ###.#  dTemp  +  ###.#  dSAL ";
TA0 * 10 ^ 6; dTAdTC; dTAdxCO2; dTAdTempC * 10 ^ 6; dTAdSAL * 10 ^ 6
            PRINT USING "                          +  ###.#  %dK0  +  ###.#  %dK1  +  ###.#  %dK2 "; dTApcdK0 *
10 ^ 6; dTApcdK1 * 10 ^ 6; dTApcdK2 * 10 ^ 6
            PRINT
            PRINT USING "   pH  =  #.####  +  #.####  dTC  +  #.####  dxCO2  +  #.####  dTemp  +  #.####  dSAL ";
pH0; dpHdTC * 10 ^ (-6); dpHdxCO2 * 10 ^ (-6); dpHdTempC; dpHdSAL
            PRINT USING "                          +  #.####  %dK0  +  #.####  %dK1  +  #.####  %dK2 "; dpHpcdK0;
dpHpcdK1; dpHpcdK2
        END IF
        TA = TA0: pH = pH0
'
'
'   ************************************************************************
    CASE 6, 12
        INPUT "   Enter pH: ", pH$: IF LEN(pH$) <> 0 THEN pH = VAL(pH$)
        LOCATE 8, 1: PRINT USING "   pH  = #.####     "; pH
        IF ICase% = 6 THEN
            INPUT "   Enter fCO2 (u-atm): ", fCO2$: IF LEN(fCO2$) <> 0 THEN fCO2 = VAL(fCO2$) * 10 ^
(-6)
            LOCATE 9, 1: PRINT USING "   fCO2 = ####.# "; fCO2 * 10 ^ 6; : PRINT SPACE$(20)
            xCO2 = fCO2 / (FugFac * VPFac)
        END IF
        IF ICase% = 12 THEN
            INPUT "   Enter xCO2 (ppm): ", xCO2$: IF LEN(xCO2$) <> 0 THEN xCO2 = VAL(xCO2$) * 10 ^
(-6)
            LOCATE 9, 1: PRINT USING "   xCO2 = ####.# "; xCO2 * 10 ^ 6; : PRINT SPACE$(20)
            fCO2 = xCO2 * FugFac * VPFac
        END IF
'
'       DO AT GIVEN pH, fCO2 OR xCO2
            H = 10 ^ (-pH)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): TC0 = TC
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): TA0 = TA
'       INCREASE pH BY dpH
            pHnew = pH + dpH
            H = 10 ^ (-pHnew)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCdpH = (TC - TC0) / dpH
            CALL FindTAfromTCpH(TC, pHnew, K(), T(), Z, TA): dTAdpH = (TA - TA0) / dpH
'       INCREASE fCO2 BY dfCO2
            fCO2new = fCO2 + dfCO2: dxCO2 = dfCO2 / (FugFac * VPFac)
            H = 10 ^ (-pH)
            TC = K0 * fCO2new * (1! + (K1 / H) * (1! + K2 / H))
            dTCdfCO2 = (TC - TC0) / dfCO2: dTCdxCO2 = (TC - TC0) / dxCO2
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA)
            dTAdfCO2 = (TA - TA0) / dfCO2: dTAdxCO2 = (TA - TA0) / dxCO2
'       INCREASE K0 BY pcdK0 %
```

```
            K0new = K0 * K0Factor
            H = 10 ^ (-pH)
            TC = K0new * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCpcdK0 = (TC - TC0) / pcdK0
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK0 = (TA - TA0) / pcdK0
'       INCREASE K1 BY pcdK1 %
            K(1) = K(1) * K1Factor: K1 = K(1)
            H = 10 ^ (-pH)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCpcdK1 = (TC - TC0) / pcdK1
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK1 = (TA - TA0) / pcdK1
            K(1) = K(1) / K1Factor: K1 = K(1)
'       INCREASE K2 BY pcdK2 %
            K(2) = K(2) * K2Factor: K2 = K(2)
            H = 10 ^ (-pH)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCpcdK2 = (TC - TC0) / pcdK2
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTApcdK2 = (TA - TA0) / pcdK2
            K(2) = K(2) / K2Factor: K2 = K(2)
'       INCREASE TempC BY dTempC
            TempCnew = TempC + dTempC
            CALL Constants(TempCnew, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            H = 10 ^ (-pH)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCdTempC = (TC - TC0) / dTempC
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdTempC = (TA - TA0) / dTempC
'       INCREASE SAL BY dSAL
            SALnew = SAL + dSAL
            CALL Constants(TempC, SALnew, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
            K1 = K(1): K2 = K(2)
            H = 10 ^ (-pH)
            TC = K0 * fCO2 * (1! + (K1 / H) * (1! + K2 / H)): dTCdSAL = (TC - TC0) / dSAL
            CALL FindTAfromTCpH(TC, pH, K(), T(), Z, TA): dTAdSAL = (TA - TA0) / dSAL
'
        LOCATE 11, 1
        IF ICase% = 6 THEN
            PRINT USING "   TA  = ####.# + ###.# dmpH + ###.# dfCO2 + ###.# dTemp + ###.# dSAL ";
TA0 * 10 ^ 6; dTAdpH * 10 ^ 3; dTAdfCO2; dTAdTempC * 10 ^ 6; dTAdSAL * 10 ^ 6
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTApcdK0 * 10
^ 6; dTApcdK1 * 10 ^ 6; dTApcdK2 * 10 ^ 6
            PRINT USING "   TC  = ####.# + ###.# dmpH + ###.# dfCO2 + ###.# dTemp + ###.# dSAL ";
TC0 * 10 ^ 6; dTCdpH * 10 ^ 3; dTCdfCO2; dTCdTempC * 10 ^ 6; dTCdSAL * 10 ^ 6
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTCpcdK0 * 10
^ 6; dTCpcdK1 * 10 ^ 6; dTCpcdK2 * 10 ^ 6
        END IF
        IF ICase% = 12 THEN
            PRINT USING "   TA  = ####.# + ###.# dmpH + ###.# dxCO2 + ###.# dTemp + ###.# dSAL ";
TA0 * 10 ^ 6; dTAdpH * 10 ^ 3; dTAdxCO2; dTAdTempC * 10 ^ 6; dTAdSAL * 10 ^ 6
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTApcdK0 * 10
^ 6; dTApcdK1 * 10 ^ 6; dTApcdK2 * 10 ^ 6
            PRINT USING "   TC  = ####.# + ###.# dmpH + ###.# dxCO2 + ###.# dTemp + ###.# dSAL ";
TC0 * 10 ^ 6; dTCdpH * 10 ^ 3; dTCdxCO2; dTCdTempC * 10 ^ 6; dTCdSAL * 10 ^ 6
            PRINT USING "                    + ###.# %dK0 + ###.# %dK1 + ###.# %dK2 "; dTCpcdK0 * 10
^ 6; dTCpcdK1 * 10 ^ 6; dTCpcdK2 * 10 ^ 6
        END IF
        TA = TA0: TC = TC0
```

```
    END SELECT
'
'
    CALL Constants(TempC, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
    CALL FindAlkContributions(pH, TC, K(), T(), BAlk, CAlk, PhosAlk, SiAlk)
    LOCATE 5, 25: PRINT USING "Phos Alk = ###.#    Boron Alk = ###.# "; PhosAlk * 10 ^ 6; BAlk * 10 ^ 6
    LOCATE 6, 25: PRINT USING "Sili Alk = ###.#    Carb Alk = ####.# "; SiAlk * 10 ^ 6; CAlk * 10 ^ 6
    LOCATE 17, 1
    PRINT "   K0 from Weiss, 1974. Estimates of its ACCURACY vary from .2% to .5%. "
    PRINT KComment1$
    PRINT KComment2$
'
'
QueryForDoingAnother:
    LOCATE 24, 1: INPUT "   ANOTHER? (<cr> = Y) ", A$
    IF LEN(A$) = 0 OR A$ = "y" OR A$ = "Y" THEN GOTO Start:
END
'*******************************************************************************
ErrorHandler:
    BEEP: BEEP
    CALL ErrorSub(ERROR$)
PrintError:
    LOCATE 20, 1: PRINT USING "   ERROR ###: "; ERR;
    PRINT ERROR$
    LOCATE 21, 5: INPUT "TYPE <cr> TO RESTART, ANYTHING ELSE TO STOP ", Q$
    IF Q$ = "" THEN
        RESUME Start:
    ELSE
        STOP
    END IF
```

```
SUB AboutThisProgram
'    THIS PRINTS THE OPENING HEADER WITH INFORMATION ON THE PROGRAM.
'
'
    CLS
    PRINT "  PROGRAM CO2SYSTM.BAS, version 1.02, written by ERNIE LEWIS "
    PRINT
    PRINT
    PRINT "  This program finds any two parameters of the CO2 system "
    PRINT "     (TA, TC, pH, and fCO2 or xCO2) from the other two. "
    PRINT "  Partials are given to show sensitivity to inputs and to K0, K1, and K2."
    PRINT
    PRINT "  xCO2 is the mole fraction of CO2 in dry air in ppm (parts per million). "
    PRINT "  fCO2 is the fugacity of CO2 in wet (H2O-saturated) air at 1 atm. "
    PRINT
    PRINT "  A choice of four sets of carbonate constants is given. "
    PRINT "  Inputs must be in umol/kg-soln and u-atm or ppm on the total pH scale. "
    PRINT "  All calculations are done in mol/kg-soln and atm on the total pH scale. "
    PRINT "  All results are in umol/kg-soln and u-atm or ppm on the total pH scale. "
    PRINT "  Phosphate and silicate are included as they contribute to the alkalinity."
    PRINT "  For more information see the program CO2SYSTM.TXT. "
    PRINT
    PRINT "  When running the program after initial inputs are made, "
    PRINT "     a <cr> at an input request will enter the last value. "
    PRINT
    PRINT "  Hit any key to continue. "
    WHILE INKEY$ = "": WEND
END SUB
```

```
SUB ChooseWhichCase (ICase%)
'    THIS PRINTS OUT THE CHOICES AND TAKES AS INPUT THE DESIRED ONE.
'    LAST REVISION OF THIS SUB 4-3-95.
'
'
    CLS : LOCATE 3, 1
    PRINT "   CHOOSE ONE OF THE FOLLOWING: "
    PRINT
    PRINT "         GIVEN          CALCULATE "
    PRINT "         -----          --------- "
    PRINT "       1) TA, TC         pH, fCO2 "
    PRINT "       2) TA, pH         TC, fCO2 "
    PRINT "       3) TA, fCO2        TC, pH "
    PRINT "       4) TC, pH         TA, fCO2 "
    PRINT "       5) TC, fCO2        TA, pH "
    PRINT "       6) pH, fCO2        TA, TC "
    PRINT
    PRINT "       7) TA, TC         pH, xCO2 "
    PRINT "       8) TA, pH         TC, xCO2 "
    PRINT "       9) TA, xCO2        TC, pH "
    PRINT "      10) TC, pH         TA, xCO2 "
    PRINT "      11) TC, xCO2        TA, pH "
    PRINT "      12) pH, xCO2        TA, TC "
'
'
MakeChoice:
    LOCATE 3, 34: INPUT "", ICase$
    IF LEN(ICase$) = 0 THEN
        LOCATE 3, 35: PRINT USING "##"; ICase%
    ELSE
        ICase% = VAL(ICase$)
    END IF
    IF ICase% < 1 OR ICase% > 12 THEN
        BEEP
        LOCATE 3, 34: PRINT "  "
        GOTO MakeChoice:
    END IF
END SUB
```

```
SUB ChooseWhichKs (WhichKs%, WhoseK$, KComment1$, KComment2$)
'    THIS ALLOWS THE USER TO CHOOSE FROM A MENU OF CARBONATE CONSTANTS.
'    LAST REVISION OF THIS SUB 4-4-95.'
     CLS
ChooseKs:
     LOCATE 5, 1
     PRINT "      CHOOSE FROM ONE OF THE FOLLOWING CHOICES FOR K1 AND K2:    "
     PRINT "      ------------------------------------------------------ "
     PRINT
     PRINT "      FOR EACH CASE, THE VALUES ARE CONVERTED (IF NECESSARY) "
     PRINT "           TO THE Total pH SCALE AND TO mol/kg-soln. "
     PRINT
     PRINT "      1) Roy, et al, Marine Chemistry 44:249-267,1993 "
     PRINT "        The 2s PRECISION of the fit is about 2% in K1 and 1.5% in K2."
     PRINT
     PRINT "      2) Goyet and Poisson, Deep-Sea Research 36:1635-1654,1989 "
     PRINT "        The 2s PRECISION of the fit is about 2.5% in K1 and 4.5% in K2. "
     PRINT
     PRINT "      3) data from Hansson, Deep-Sea Research 20:461-478,1973 "
     PRINT "        refit by Dickson and Millero, DSR, 34:1733-1743,1987 "
     PRINT "        The 2s PRECISION of the fit is about 3% in K1 and 4% in K2. "
     PRINT
     PRINT "      4) data from Mehrbach et al, Limn. and Ocean., 18:897-907,1973 "
     PRINT "        refit by Dickson and Millero, DSR, 34:1733-1743,1987 "
     PRINT "        The 2s PRECISION of the fit is about 2.5% in K1 and 4.5% in K2. "
"
     LOCATE 5, 66: INPUT "", WhichKs$
     IF LEN(WhichKs$) <> 0 THEN WhichKs% = VAL(WhichKs$)
     IF WhichKs% < 1 OR WhichKs% > 4 THEN
          BEEP
          GOTO ChooseKs:
     END IF
'
GetKComments:
     SELECT CASE WhichKs%
          CASE 1
               WhoseK$ = "ROY et al"
               KComment1$ = "   Carbonate constants of Roy, 1993. "
               KComment2$ = "   The 2s PRECISION of the fit is about 2% in K1 and 1.5% in K2."
          CASE 2
               WhoseK$ = "GOYET, POISSON"
               KComment1$ = "   Carbonate constants of Goyet and Poisson, 1989. "
               KComment2$ = "   The 2s PRECISION of the fit is about 2.5% in K1 and 4.5% in K2. "
          CASE 3
               WhoseK$ = "HANSSON"
               KComment1$ = "   Carbonate constants from data of Hansson, refit by Dickson and Millero. "
               KComment2$ = "   The 2s PRECISION of the fit is about 3% in K1 and 4% in K2. "
          CASE 4
               WhoseK$ = "MEHRBACH et al"
               KComment1$ = "   Carbonate constants from data of Mehrbach, refit by Dickson and Millero "
               KComment2$ = "   The 2s PRECISION of the fit is about 2.5% in K1 and 4.5% in K2. "
     END SELECT
END SUB
```

```
SUB Constants (TempC, SAL, PRES, D, K0, WhichKs%, K(), T(), Z, NERNST, FugFac, VPFac)
'    THIS WILL CALCULATE CONSTANTS FOR THE CO2 SYSTEM AND ALKALINITY.
'    ALL THE LOGS HERE ARE BASE e.
'    WRITTEN BY ERNIE LEWIS.
'    LAST REVISION 5-10-95.
'
'
    TempK = TempC + 273.15
'
'
'*************************************************
CalculateDH20:
    D0 = .999842594# + 6.793952E-05 * TempC - 9.09529E-06 * TempC ^ 2
    D0 = D0 + 1.001685E-07 * TempC ^ 3 - 1.120083E-09 * TempC ^ 4
    D0 = D0 + 6.536332E-12 * TempC ^ 5
'
'
'*************************************************
CalculateDSW:
    A = 8.24493E-04 - 4.0899E-06 * TempC + 7.6438E-08 * TempC ^ 2
    A = A - 8.2467E-10 * TempC ^ 3 + 5.3875E-12 * TempC ^ 4
    B = -5.72466E-06 + 1.0227E-07 * TempC - 1.6546E-09 * TempC ^ 2
    C = 4.8314E-07
    D = D0 + (A * SAL) + (B * SAL ^ (3 / 2)) + (C * SAL ^ 2)
'
'
'*************************************************
CalculateTs:
'    THESE ARE IN mol/kg-soln
    TB = (.000232 / 10.811) * (SAL / 1.80655)
    TF = (.000067 / 18.998) * (SAL / 1.80655)
    TS = (.14 / 96.062) * (SAL / 1.80655)
'
'
'*************************************************
MakeTMatrix:
'    NOW PUT TOTAL VALUES IN T MATRIX
    T(1) = TB
    T(2) = TF
    T(3) = TS
'    T(4) = TP
'    T(5) = TSi
'    THESE LAST TWO WERE SET EARLIER
'
'
'*************************************************
CalculateKs:
    IonS = 19.924 * SAL / (1000! - 1.005 * SAL)
'
'
    LNK0 = -60.2409 + 93.4517 * (100! / TempK) + 23.3585 * LOG(TempK / 100!)
    LNK0 = LNK0 + SAL * (.023517 - .023656 * TempK / 100! + .0047036 * (TempK / 100!) ^ 2)
    K0 = EXP(LNK0)
```

```
'
'
    LNKBTOP = -8966.9 - 2890.53 * SQR(SAL) - 77.942 * SAL
    LNKBTOP = LNKBTOP + 1.728 * SAL ^ (3 / 2) - .0996 * SAL ^ 2
    LNKB = LNKBTOP / TempK
    LNKB = LNKB + 148.0248 + 137.1942 * SQR(SAL) + 1.62142 * SAL
    LNKB = LNKB + (-24.4344 - 25.085 * SQR(SAL) - .2474 * SAL) * LOG(TempK)
    LNKB = LNKB + .053105 * SQR(SAL) * TempK
    KB = EXP(LNKB)
'
'
    LNKS = -4276.1 / TempK + 141.328 - 23.093 * LOG(TempK)
    LNKS = LNKS + (-13856 / TempK + 324.57 - 47.986 * LOG(TempK)) * SQR(IonS)
    LNKS = LNKS + (35474 / TempK - 771.54 + 114.723 * LOG(TempK)) * IonS
    LNKS = LNKS - (2698 / TempK) * (IonS) ^ (3 / 2) + (1776 / TempK) * IonS ^ 2
    LNKS = LNKS + LOG(1! - .001005 * SAL): ' convert to mol/kg-soln
    KS = EXP(LNKS)
'
'
    FREEtoTOT = (1! + TS / KS): ' pH scale conversion factor
'
'
    LNKF = 1590.2 / TempK - 12.641 + 1.525 * SQR(IonS)
    LNKF = LNKF + LOG(1! - .001005 * SAL): ' convert to mol/kg-soln
    KFfree = EXP(LNKF)
    KF = KFfree * FREEtoTOT: ' convert from free to total pH scale
'
'
    SWStoTOT = (1! + TS / KS) / (1! + TS / KS + TF / KFfree): ' pH scale conversion factor
'
'
    LNKW = 148.9802 - 13847.26 / TempK - 23.6521 * LOG(TempK)
    LNKW = LNKW + (-5.977 + 118.67 / TempK + 1.0495 * LOG(TempK)) * SQR(SAL)
    LNKW = LNKW - .01615 * SAL
    KW = EXP(LNKW)
    KW = KW * SWStoTOT: ' convert from SWS to total pH scale
'
'
    LNKP1 = -4576.752 / TempK + 115.54 - 18.453 * LOG(TempK)
    LNKP1 = LNKP1 + (-106.736 / TempK + .69171) * SQR(SAL)
    LNKP1 = LNKP1 + (-.65643 / TempK - .01844) * SAL
    KP1 = EXP(LNKP1)
    KP1 = KP1 * SWStoTOT: ' convert from SWS to total pH scale
'
'
    LNKP2 = -8814.715 / TempK + 172.1033 - 27.927 * LOG(TempK)
    LNKP2 = LNKP2 + (-160.34 / TempK + 1.3566) * SQR(SAL)
    LNKP2 = LNKP2 + (.37335 / TempK - .05778) * SAL
    KP2 = EXP(LNKP2)
    KP2 = KP2 * SWStoTOT: ' convert from SWS to total pH scale
'
'
    LNKP3 = -3070.75 / TempK - 18.126
```

```
    LNKP3 = LNKP3 + (17.27039 / TempK + 2.81197) * SQR(SAL)
    LNKP3 = LNKP3 + (-44.99486 / TempK - .09984) * SAL
    KP3 = EXP(LNKP3)
    KP3 = KP3 * SWStoTOT: ' convert from SWS to total pH scale
'
'
    LNKSi = -8904.2 / TempK + 117.4 - 19.334 * LOG(TempK)
    LNKSi = LNKSi + (-458.79 / TempK + 3.5913) * SQR(IonS)
    LNKSi = LNKSi + (188.74 / TempK - 1.5998) * IonS
    LNKSi = LNKSi + (-12.1652 / TempK + .07871) * IonS ^ 2
    LNKSi = LNKSi + LOG(1! - .001005 * SAL): ' convert to mol/kg-soln
    KSi = EXP(LNKSi)
    KSi = KSi * SWStoTOT: ' convert from SWS to total pH scale
'
'
'*************************************************
'*************************************************
CalculateCarbonateKs:
'   ALL CALCULATIONS ARE DONE ON THE TOTAL pH SCALE IN mol/kg-soln.
'
'
SELECT CASE WhichKs%
'*************************************************
    CASE 1: ' ROY et al, Marine Chemistry, 44:249-267, 1993
'       K1, K2 are on the Total pH scale and given in mol/kg-H20.
'       To convert to mol/kg-soln, multiply their Ki by (1. - .001005 * SAL)
'       There is a typo in the abstract on p. 249: in the eq. for lnK1* the last
'           term should have S raised to the power 1.5
'       There is a typo on p. 254 in the label for Table 2
'           the coefficient of m(CO3 2-) should be a 2, not a 3
'       Tables 3 and 4 do not match either fit given, but refer to the calculation
'           of Ki from the EMF data. This was not clear from the titles.
'       Andrew Dickson (personal communication 12-1-94) says that he has been
'       sent revisions of tables 3 and 4.
'       The standard deviation of the fit in lnK1 is .0048, which is .5% in K1.
'       They claim a 2s precision of .004 in pK1 (p. 258). This is consistent.
'       The standard deviation of the fit in lnK2 is .007, which is .7% in K2.
'       They claim a 2s precision of .006 in pK2 (p. 258). This is consistent.
'       These seem to be backwards, from looking at Fig.3 and Fig.4 on p. 255.
'       Millero (1995) states an rms deviation of about .004 in pK1 and .003 in pK2.
'       This would correspond to about 2% in K1 and 1.5% in K2 for a 2s precision.
'       This is eq. 29 on p. 254 and what they use in their abstract.
    lnK1 = 2.83655 - 2307.1266# / TempK - 1.5529413# * LOG(TempK)
    lnK1 = lnK1 + (-.20760841# - 4.0484 / TempK) * SQR(SAL)
    lnK1 = lnK1 + 8.468345E-02 * SAL - .00654208# * SAL ^ (1.5)
    K1 = EXP(lnK1)
    K1 = K1 * (1! - .001005 * SAL): ' convert to mol/kg-soln
'
'       This is equation 30 on p. 254 and what they use in their abstract.
    lnK2 = -9.226508 - 3351.6106# / TempK - .2005743 * LOG(TempK)
    lnK2 = lnK2 + (-.106901773# - 23.9722 / TempK) * SQR(SAL)
    lnK2 = lnK2 + .1130822 * SAL - 8.46934E-03 * SAL ^ (1.5)
    K2 = EXP(lnK2)
```

```
    K2 = K2 * (1! - .001005 * SAL): ' convert to mol/kg-soln
'
'*************************************************
    CASE 2: ' GOYET AND POISSON, Deep-Sea Research, 36(11):1635-1654, 1989
'       K1, K2 are on the SWS pH scale and in mol/kg-soln.
'       This is in Table 5 on p. 1652 and what they use in the abstract.
'       The goodness of fit (2s) in pK1 is .011, or 2.5% in K1.
    pK1 = 812.27 / TempK + 3.356 - .00171 * SAL * LOG(TempK)
    pK1 = pK1 + .000091 * SAL ^ 2
    K1 = 10 ^ (-pK1)
    K1 = K1 * SWStoTOT: ' convert from SWS to total pH scale
'
'       This is in Table 5 on p. 1652 and what they use in the abstract.
'       The goodness of fit (2s) in pK2 is .02, or 4.5% in K2.
    pK2 = 1450.87 / TempK + 4.604 - .00385 * SAL * LOG(TempK)
    pK2 = pK2 + .000182 * SAL ^ 2
    K2 = 10 ^ (-pK2)
    K2 = K2 * SWStoTOT: ' convert from SWS to total pH scale
'
'*************************************************
    CASE 3: ' HANSSON refit BY DICKSON AND MILLERO
'       K1, K2 from data of Hansson, Deep-Sea Research, 20:461-478, 1973
'       and Hansson, Acta Chemica Scandanavia, 27, 931-944, 1973.
'       He gave his results on the Total scale (he called it
'           the seawater scale) and in mol/kg-soln.
'       Dickson and Millero, Deep-Sea Research, 34(10):1733-1743, 1987
'       (see also Corrigenda, Deep-Sea Research, 36:983, 1989)
'           refit his data on the SWS pH scale in mol/kg-soln.
'       !!!!NOTE: There is a typo in DM on p. 1739 in Table 4.
'       !!!! The equation for pK2* for Hansson should have a .000132 *S^2
'           instead of a .000116 *S^2.
'
'       This is from Table 4 on p. 1739.
'       The precision in pK1 is .013, or 3% in K1.
    pK1 = 851.4 / TempK + 3.237 - .0106 * SAL + .000105 * SAL ^ 2
    K1 = 10 ^ (-pK1)
    K1 = K1 * SWStoTOT: ' convert from SWS to total pH scale
'
'       This is from Table 4 on p. 1739.
'       The precision in pK2 is .017, or 4.1% in K2.
    pK2 = -3885.4 / TempK + 125.844 - 18.141 * LOG(TempK)
    pK2 = pK2 - .0192 * SAL + .000132 * SAL ^ 2
    K2 = 10 ^ (-pK2)
    K2 = K2 * SWStoTOT: ' convert from SWS to total pH scale
'
'*************************************************
    CASE 4: ' MEHRBACH refit BY DICKSON AND MILLERO
'       K1, K2 from Mehrbach et al Limnol. Oceanogr.,18(6):897-907, 1973
'       They worked on the NBS scale and gave their results in mol/kg-soln.
'       Dickson and Millero, Deep-Sea Research,34(10):1733-1743, 1987
'           refit their data on the SWS pH scale in mol/kg-soln.
'   This is in Table 4 on p. 1739.
```

```
'     The precision (2 sd) in pK1 is .011, or 2.6% in K1.
      pK1 = 3670.7 / TempK - 62.008 + 9.7944 * LOG(TempK)
      pK1 = pK1 - .0118 * SAL + .000116 * SAL ^ 2
      K1 = 10 ^ (-pK1)
      K1 = K1 * SWStoTOT: ' convert from SWS to total pH scale
'
'     This is in Table 4 on p. 1739.
'     The precision (2 sd) in pK2 is .020, or 4.6% in K2.
      pK2 = 1394.7 / TempK + 4.777 - .0184 * SAL + .000118 * SAL ^ 2
      K2 = 10 ^ (-pK2)
      K2 = K2 * SWStoTOT: ' convert from SWS to total pH scale
'
'
'*************************************************
END SELECT
'
'
'*************************************************
'*************************************************
MakeKMatrix:
'     NOW PUT EQUILIBRIUM CONSTANTS IN K MATRIX
      K(1) = K1
      K(2) = K2
      K(3) = KW
      K(4) = KB
      K(5) = KF
      K(6) = KS
      K(7) = KP1
      K(8) = KP2
      K(9) = KP3
      K(10) = KSi
'
'
'*************************************************
CalculateOtherConstants:
      Z = 1 + TS / KS
      R = 8.31451:  ' IN N-m/(mol-K)
      F = 96.485309#: ' IN N-m/(mol-mV)
      NERNST = R * TempK * LOG(10) / F
'
'
'*************************************************
CalculateVPFac:
' Weiss, R. F., and Price, B. A., Nitrous oxide solubility in water and
'     seawater, Marine Chemistry 8, 347-359, 1980.
' They fit the data of Goff and Gratch (1946) with the vapor pressure
'     lowering by sea salt as given by Robinson (1954).
' This fits the more complicated Goff and Gratch, and Robinson equations
'     from 273 to 313 deg K and 0 to 40 SAL with a standard error
'     of .015%, about 5 * 10^(-6) atm over this range.
' This may be on IPTS-29 since they didn't mention the temperature scale,
'     and the data of Goff and Gratch came before IPTS-48.
' The references are:
```

```
' Goff, J. A. and Gratch, S., Low pressure properties of water from -160 deg
'     to 212 deg F, Transactions of the American Society of Heating and
'     Ventilating Engineers 52, 95-122, 1946.
' Robinson, Journal of the Marine Biological Association of the U. K. 33,
'     449-455, 1954.
'
'
'     This is eq. 10 on p. 350.
'     log refers to log base e.
'     This is in atmospheres.
      VPWP = EXP(24.4543 - 67.4509 * (100! / TempK) - 4.8489 * LOG(TempK / 100!))
      VPCorrWP = EXP(-.000544 * SAL)
      VPSWWP = VPWP * VPCorrWP
      VPFac = PRES - VPSWWP
'
'
'*************************************************
CalculateFugacityConstants:
      DELTA = (57.7 - .118 * TempK) * 10 ^ (-6): ' IN m^3 / mol
'
'
      B = -1636.75 + 12.0408 * TempK - .0327957 * TempK ^ 2
      B = B + 3.16528 * 10 ^ (-5) * TempK ^ 3
      B = B * 10 ^ (-6): ' IN m3 / mol
'
'
'     FOR A MIXTURE OF CO2 AND AIR at 1 atm (at low CO2 concentrations):
      PRESinPa = PRES * 101325!
      EXPONENT = ((B + 2! * DELTA) * PRESinPa / (R * TempK))
      FugFac = EXP(EXPONENT)
END SUB
```

```
SUB ErrorSub (ERROR$)
'    WRITTEN BY ERNIE LEWIS
'    MOST RECENT VERSION 11-28-94
'
'

    IF ERR = 1 THEN ERROR$ = "NEXT WITHOUT FOR ***** "
    IF ERR = 2 THEN ERROR$ = "SYNTAX ERROR ***** "
    IF ERR = 3 THEN ERROR$ = "RETURN WITHOUT GOSUB ***** "
    IF ERR = 4 THEN ERROR$ = "OUT OF DATA ***** "
    IF ERR = 5 THEN ERROR$ = "ILLEGAL FUNCTION CALL ***** "
    IF ERR = 6 THEN ERROR$ = "OVERFLOW ***** "
    IF ERR = 7 THEN ERROR$ = "OUT OF MEMORY ***** "
    IF ERR = 8 THEN ERROR$ = "LABEL NOT DEFINED**** "
    IF ERR = 9 THEN ERROR$ = "SUBSCRIPT OUT OF RANGE ***** "
    IF ERR = 10 THEN ERROR$ = "DUPLICATE DEFINITION ***** "
    IF ERR = 11 THEN ERROR$ = "DIVISION BY ZERO ***** "
    IF ERR = 12 THEN ERROR$ = "ILLEGAL IN DIRECT MODE ***** "
    IF ERR = 13 THEN ERROR$ = "TYPE MISMATCH ***** "
    IF ERR = 14 THEN ERROR$ = "OUT OF STRING SPACE ***** "
    IF ERR = 16 THEN ERROR$ = "STRING FORMULA TOO COMPLEX ***** "
    IF ERR = 17 THEN ERROR$ = "CANNOT CONTINUE ***** "
    IF ERR = 18 THEN ERROR$ = "FUNCTION NOT DEFINED ***** "
    IF ERR = 19 THEN ERROR$ = "NO RESUME ***** "
    IF ERR = 20 THEN ERROR$ = "RESUME WITHOUT ERROR ***** "
    IF ERR = 24 THEN ERROR$ = "DEVICE TIMEOUT ***** "
    IF ERR = 25 THEN ERROR$ = "DEVICE FAULT ***** "
    IF ERR = 26 THEN ERROR$ = "FOR WITHOUT NEXT ***** "
    IF ERR = 27 THEN ERROR$ = "OUT OF PAPER ***** "
    IF ERR = 29 THEN ERROR$ = "WHILE WITHOUT WEND ***** "
    IF ERR = 30 THEN ERROR$ = "WEND WITHOUT WHILE ***** "
    IF ERR = 33 THEN ERROR$ = "DUPLICATE LABEL ***** "
    IF ERR = 35 THEN ERROR$ = "SUBPROGRAM NOT DEFINED ***** "
    IF ERR = 37 THEN ERROR$ = "ARGUMENT-COUNT MISMATCH ***** "
    IF ERR = 38 THEN ERROR$ = "ARRAY NOT DEFINED ***** "
    IF ERR = 40 THEN ERROR$ = "VARIABLE REQUIRED ***** "
    IF ERR = 50 THEN ERROR$ = "FIELD OVERFLOW ***** "
    IF ERR = 51 THEN ERROR$ = "INTERNAL ERROR ***** "
    IF ERR = 52 THEN ERROR$ = "BAD FILENAME OR NUMBER ***** "
    IF ERR = 53 THEN ERROR$ = "FILE NOT FOUND** ***** "
    IF ERR = 54 THEN ERROR$ = "BAD FILE MODE ***** "
    IF ERR = 55 THEN ERROR$ = "FILE ALREADY OPEN ***** "
    IF ERR = 56 THEN ERROR$ = "FIELD STATEMENT ACTIVE ***** "
    IF ERR = 57 THEN ERROR$ = "DEVICE I/O ERROR ***** "
    IF ERR = 58 THEN ERROR$ = "FILE ALREADY EXISTS ***** "
    IF ERR = 59 THEN ERROR$ = "BAD RECORD LENGTH ***** "
    IF ERR = 61 THEN ERROR$ = "DISK FULL ***** "
    IF ERR = 62 THEN ERROR$ = "INPUT PAST END OF FILE ***** "
    IF ERR = 63 THEN ERROR$ = "BAD RECORD NUMBER ***** "
    IF ERR = 64 THEN ERROR$ = "BAD FILENAME ***** "
    IF ERR = 67 THEN ERROR$ = "TOO MANY FILES ***** "
    IF ERR = 68 THEN ERROR$ = "DEVICE UNAVAILABLE ***** "
    IF ERR = 69 THEN ERROR$ = "COMMUNICATION BUFFER OVERFLOW ***** "
    IF ERR = 70 THEN ERROR$ = "PERMISSION DENIED ***** "
```

```
    IF ERR = 71 THEN ERROR$ = "DISK NOT READY ***** "
    IF ERR = 72 THEN ERROR$ = "DISK-MEDIA ERROR--BAD FORMAT?**** "
    IF ERR = 73 THEN ERROR$ = "FEATURE UNAVAILABLE ***** "
    IF ERR = 74 THEN ERROR$ = "RENAME ACROSS DISKS ***** "
    IF ERR = 75 THEN ERROR$ = "PATH / FILE ACCESS ERROR ***** "
    IF ERR = 76 THEN ERROR$ = "PATH NOT FOUND ***** "
    IF ERR = 80 THEN ERROR$ = "FEATURE REMOVED ***** "
    IF ERR = 81 THEN ERROR$ = "INVALID NAME ***** "
    IF ERR = 82 THEN ERROR$ = "TABLE NOT FOUND ***** "
    IF ERR = 83 THEN ERROR$ = "INDEX NOT FOUND ***** "
    IF ERR = 84 THEN ERROR$ = "INVALID COLUMN ***** "
    IF ERR = 85 THEN ERROR$ = "NO CURRENT RECORD ***** "
    IF ERR = 86 THEN ERROR$ = "DUPLICATE VALUE FOR UNIQUE INDEX ***** "
    IF ERR = 87 THEN ERROR$ = "INVALID OPERATION ON NULL INDEX ***** "
    IF ERR = 88 THEN ERROR$ = "DATABASE NEEDS REPAIR ***** "
END SUB
```

```
SUB FindAlkContributions (pH, TC, K(), T(), BAlk, CAlk, PhosAlk, SiAlk)
'    THIS CALCULATES THE CONTRIBUTIONS TO THE ALKALINITY FROM BORON,
'        CARBONATE ( AND BICARBONATE), PHOSPHATE, AND SILICATE.
'    LAST REVISION OF THIS SUB 4-4-95.
'
'
'*****************************************
'    NOW GET EQUILIBRIUM CONSTANTS FROM K MATRIX
     K1 = K(1)
     K2 = K(2)
     KW = K(3)
     KB = K(4)
     KF = K(5)
     KS = K(6)
     KP1 = K(7)
     KP2 = K(8)
     KP3 = K(9)
     KSi = K(10)
'
'
'*************************************
'    NOW GET TOTAL VALUES FROM T MATRIX
     TB = T(1)
     TP = T(4)
     TSi = T(5)
'
'
'*************************************
     H = 10 ^ (-pH)
     CAlk = TC * K1 * (H + 2! * K2) / (H * H + K1 * H + K1 * K2)
     OH = KW / H
     BAlk = TB / (1! + H / KB)
          PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
          PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
     PhosAlk = TP * PhosTOP / PhosBOT
     SiAlk = TSi / (1! + H / KSi)
END SUB



SUB FindfCO2fromTCpH (TC, pH, K0, K1, K2, fCO2)
'    THIS FINDS THE VALUE OF fCO2 FROM TC AND pH, USING K0, K1, AND K2.
'    LAST REVISION OF THIS SUB 1-1-95.
'
'
     H = 10 ^ (-pH)
     HCO3 = TC * K1 * H / (H * H + K1 * H + K1 * K2)
     CO2Star = H * HCO3 / K1
     fCO2 = CO2Star / K0
END SUB
```

```
SUB FindpHfromTAfCO2 (TA, fCO2, K0, K(), T(), Z, pH)
'    THIS FINDS THE VALUE OF [H+] FROM TA AND fCO2 USING K0, K1, K2.
'    FOR A STARTING GUESS, IT IS ASSUMED THAT ALL Alk IS CAlk.
'    THIS WILL ALWAYS PUT pH HIGH, WHICH IS BETTER FOR CONVERGENCE.
'    NEWTON'S METHOD IS USED.
'    LAST REVISION OF THIS SUB 1-1-95.
'
'
'*****************************************
'    NOW GET EQUILIBRIUM CONSTANTS FROM K MATRIX
     K1 = K(1)
     K2 = K(2)
     KW = K(3)
     KB = K(4)
     KF = K(5)
     KS = K(6)
     KP1 = K(7)
     KP2 = K(8)
     KP3 = K(9)
     KSi = K(10)
'
'
'*************************************
'    NOW GET TOTAL VALUES FROM T MATRIX
     TB = T(1)
     TF = T(2)
     TS = T(3)
     TP = T(4)
     TSi = T(5)
'
'
'*************************************
InitializeH2:
     BB = -K0 * K1 * fCO2 / TA
     CC = -2! * K0 * K1 * K2 * fCO2 / TA
     HGuess = -.5 * BB + .5 * SQR(BB ^ 2 - 4! * CC)
     deltaH = 10 ^ (-9)
     HStart = HGuess + deltaH
     TOL = 10 ^ (-4): ' THIS IS 10^(-4) / LOG(10) pH UNITS
'
'
'************************************************
     H = HStart
     HCO3 = K0 * K1 * fCO2 / H
     CO3 = K0 * K1 * K2 * fCO2 / (H * H)
     CAlk = HCO3 + 2! * CO3
     HFree = H / Z
     OH = KW / H
     BAlk = TB / (1! + H / KB)
     HSO4 = TS * H / (H + KS * Z)
     HF = TF * H / (H + KF)
          PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
          PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
```

```
    PhosAlk = TP * PhosTOP / PhosBOT
    SiAlk = TSi / (1! + H / KSi)
    Residual = TA - CAlk + HFree - OH + HS04 + HF - BAlk - PhosAlk - SiAlk
'
'
'*************************************************
    H = HGuess
Newton2:
    ResidualPlus = Residual
    HCO3 = K0 * K1 * fCO2 / H
    CO3 = K0 * K1 * K2 * fCO2 / (H * H)
    CAlk = HCO3 + 2! * CO3
    HFree = H / Z
    OH = KW / H
    BAlk = TB / (1! + H / KB)
    HSO4 = TS * H / (H + KS * Z)
    HF = TF * H / (H + KF)
        PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
        PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
    PhosAlk = TP * PhosTOP / PhosBOT
    SiAlk = TSi / (1! + H / KSi)
    Residual = TA - CAlk + HFree - OH + HS04 + HF - BAlk - PhosAlk - SiAlk
'
'
    Slope = (ResidualPlus - Residual) / deltaH
    deltaH = Residual / Slope
    H = H - deltaH
    IF ABS(deltaH) > TOL * H THEN GOTO Newton2:
    pH = LOG(H) / LOG(.1)
END SUB
```

```
SUB FindpHfromTATC (TA, TC, K(), T(), Z, pH)
'    THIS FINDS THE pH FROM TA AND TC USING K1 AND K2.
'    FOR FIRST GUESS, IT IS ASSUMED THAT ALL Alk IS CAlk.
'    THIS WILL ALWAYS PUT pH HIGH, WHICH IS BETTER FOR CONVERGENCE.
'    NEWTON'S METHOD IS USED.
'    LAST REVISION OF THIS SUB 1-1-95.
'
'
'*****************************************************
'    NOW GET EQUILIBRIUM CONSTANTS FROM K MATRIX
     K1 = K(1)
     K2 = K(2)
     KW = K(3)
     KB = K(4)
     KF = K(5)
     KS = K(6)
     KP1 = K(7)
     KP2 = K(8)
     KP3 = K(9)
     KSi = K(10)
'
'
'*****************************************************
'    NOW GET TOTAL VALUES FROM T MATRIX
     TB = T(1)
     TF = T(2)
     TS = T(3)
     TP = T(4)
     TSi = T(5)
'
'
'*****************************************************
InitializeH1:
     BB = (1! - TC / TA) * K1
     CC = (1! - 2! * TC / TA) * K1 * K2
     HGuess = -.5 * BB + .5 * SQR(BB ^ 2 - 4! * CC)
     deltaH = 10 ^ (-9)
     HStart = HGuess + deltaH
     TOL = 10 ^ (-4): ' THIS IS 10^(-4) / LOG(10) pH UNITS
'
'
     H = HStart
     CAlk = TC * K1 * (H + 2! * K2) / (H * H + K1 * H + K1 * K2)
     HFree = H / Z
     OH = KW / H
     BAlk = TB / (1! + H / KB)
     HSO4 = TS * H / (H + KS * Z)
     HF = TF * H / (H + KF)
          PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
          PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
     PhosAlk = TP * PhosTOP / PhosBOT
     SiAlk = TSi / (1! + H / KSi)
     Residual = TA - CAlk + HFree - OH + HS04 + HF - BAlk - PhosAlk - SiAlk
```

```
'
'
    H = HGuess
NEWTON1:
    ResidualPlus = Residual
    CAlk = TC * K1 * (H + 2! * K2) / (H * H + K1 * H + K1 * K2)
    HFree = H / Z
    OH = KW / H
    BAlk = TB / (1! + H / KB)
    HSO4 = TS * H / (H + KS * Z)
    HF = TF * H / (H + KF)
        PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
        PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
    PhosAlk = TP * PhosTOP / PhosBOT
    SiAlk = TSi / (1! + H / KSi)
    Residual = TA - CAlk + HFree - OH + HSO4 + HF - BAlk - PhosAlk - SiAlk
'
'
    Slope = (ResidualPlus - Residual) / deltaH
    deltaH = Residual / Slope
    H = H - deltaH
    IF (ABS(deltaH) > TOL * H) GOTO NEWTON1
    pH = LOG(H) / LOG(.1)
END SUB
```

```
SUB FindpHfromTCfCO2 (TC, fCO2, K0, K1, K2, pH)
'   THIS FINDS pH FROM TC AND fCO2, USING K0, K1, AND K2.
'   FOR DERIVATIONS SEE SOP version 2.0, p. 12/14 OF Chapter 2.
'   LAST REVISION OF THIS SUB 1-1-95.
'
'
    CO2Star = K0 * fCO2
    K = K1 / K2
    TERM1 = K * K0 * fCO2
    HCO3 = -.5 * TERM1 + SQR(TERM1 * TERM1 - 4! * TERM1 * (K0 * fCO2 - TC)) / 2!
    CO3 = TC - CO2Star - HCO3
    H = K1 * CO2Star / HCO3
    pH = LOG(H) / LOG(.1)
END SUB
```

```
SUB FindTAfromTCpH (TC, pH, K(), T(), Z, TA)
'    THIS FINDS TA GIVEN TC AND pH.
'    LAST REVISION OF THIS SUB 1-1-95.
'
'
'*************************************************
'    NOW GET EQUILIBRIUM CONSTANTS FROM K MATRIX
     K1 = K(1)
     K2 = K(2)
     KW = K(3)
     KB = K(4)
     KF = K(5)
     KS = K(6)
     KP1 = K(7)
     KP2 = K(8)
     KP3 = K(9)
     KSi = K(10)
'
'
'*************************************************
'    NOW GET TOTAL VALUES FROM T MATRIX
     TB = T(1)
     TF = T(2)
     TS = T(3)
     TP = T(4)
     TSi = T(5)
'
'
'*************************************************
     H = 10 ^ (-pH)
     CAlk = TC * K1 * (H + 2! * K2) / (H * H + K1 * H + K1 * K2)
     HFree = H / Z
     OH = KW / H
     BAlk = TB / (1! + H / KB)
     HSO4 = TS * H / (H + KS * Z)
     HF = TF * H / (H + KF)
          PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
          PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
     PhosAlk = TP * PhosTOP / PhosBOT
     SiAlk = TSi / (1! + H / KSi)
     TA = CAlk - HFree + OH - HSO4 - HF + BAlk + PhosAlk + SiAlk
END SUB
```

```
SUB FindTCfromTApH (TA, pH, K(), T(), Z, TC)
'    THIS FINDS THE TC FROM TA AND pH.
'    LAST REVISION OF THIS SUB 1-1-95.
'
'
'*****************************************************
'    NOW GET EQUILIBRIUM CONSTANTS FROM K MATRIX
     K1 = K(1)
     K2 = K(2)
     KW = K(3)
     KB = K(4)
     KF = K(5)
     KS = K(6)
     KP1 = K(7)
     KP2 = K(8)
     KP3 = K(9)
     KSi = K(10)
'
'
'*****************************************************
'    NOW GET TOTAL VALUES FROM T MATRIX
     TB = T(1)
     TF = T(2)
     TS = T(3)
     TP = T(4)
     TSi = T(5)
'
'
'*****************************************************
     H = 10 ^ (-pH)
     HFree = H / Z
     OH = KW / H
     BAlk = TB / (1! + H / KB)
     HSO4 = TS * H / (H + KS * Z)
     HF = TF * H / (H + KF)
          PhosTOP = KP1 * KP2 * H + 2! * KP1 * KP2 * KP3 - H ^ 3
          PhosBOT = H ^ 3 + KP1 * H ^ 2 + KP1 * KP2 * H + KP1 * KP2 * KP3
     PhosAlk = TP * PhosTOP / PhosBOT
     SiAlk = TSi / (1! + H / KSi)
     CAlk = TA + HFree - OH + HSO4 + HF - BAlk - PhosAlk - SiAlk
'
'
     CO2Star = CAlk * H * H / (K1 * (H + 2! * K2))
     HCO3 = CAlk * H / (H + 2! * K2)
     CO3 = CAlk * K2 / (H + 2! * K2)
     TC = CO2Star + HCO3 + CO3
END SUB
```

```
SUB InputStuff (ICase%, TempC, SAL, PRES, TP, TSi)
'
'
    CLS
    SELECT CASE ICase%
        CASE 1, 2, 3, 4, 5, 6
            PRINT "   TA, TC in umol/kg-soln   fCO2 in u-atm   pH on the Total scale "
        CASE 7, 8, 9, 10, 11, 12
            PRINT "   TA, TC in umol/kg-soln   xCO2 in ppm    pH on the Total scale "
    END SELECT
'
'
InputTemp:
    LOCATE 3, 1: INPUT "   Enter TempC: ", TempC$
    IF LEN(TempC$) <> 0 THEN TempC = VAL(TempC$)
    LOCATE 3, 1: PRINT USING "   TempC = ##.## "; TempC; : PRINT SPACE$(20)
'
'
InputSal:
    LOCATE 4, 1: INPUT "   Enter SAL: ", SAL$
    IF LEN(SAL$) <> 0 THEN SAL = VAL(SAL$)
    LOCATE 4, 1: PRINT USING "   SAL   = ##.## "; SAL; : PRINT SPACE$(20)
'
'
InputPressure:
    PRES = 1!
'
'
InputPhosphate:
    LOCATE 5, 1: INPUT "   Enter total phosphate (umol/kg-soln): ", TP$
    IF LEN(TP$) <> 0 THEN TP = VAL(TP$)
    LOCATE 5, 1: PRINT USING "   Phos = ###.# "; TP; : PRINT SPACE$(40)
'
'
InputSilicate:
    LOCATE 6, 1: INPUT "   Enter total silicate (umol/kg-soln): ", TSi$
    IF LEN(TSi$) <> 0 THEN TSi = VAL(TSi$)
    LOCATE 6, 1: PRINT USING "   Sili = ###.# "; TSi; : PRINT SPACE$(40)
'
'
    LOCATE 8, 1
END SUB
```

```
SUB SetParameters (dTA, dTC, dpH, dfCO2, dTempC, dSAL, pcdK0, pcdK1, pcdK2, K0Factor, K1Factor, K2Factor)
'    THIS SETS ALL PARAMETERS WHICH ARE USED
'
'
    dTA = 10 ^ (-6):      ' 1 umol/kg-soln
    dTC = 10 ^ (-6):      ' 1 umol/kg-soln
    dpH = .001:           ' 1 mpH on TOTAL pH scale
    dfCO2 = 10 ^ (-6):    ' 1 u-atm
    dTempC = 1:           ' 1 deg C
    dSAL = 1:             ' 1 mille
    pcdK0 = 1:            ' 1% umol/kg-soln/atm
    pcdK1 = 1:            ' 1% on TOTAL pH scale in mol/kg-soln
    pcdK2 = 1:            ' 1% on TOTAL pH scale in mol/kg-soln
    K0Factor = 1! + pcdK0 / 100!
    K1Factor = 1! + pcdK1 / 100!
    K2Factor = 1! + pcdK2 / 100!
END SUB
```