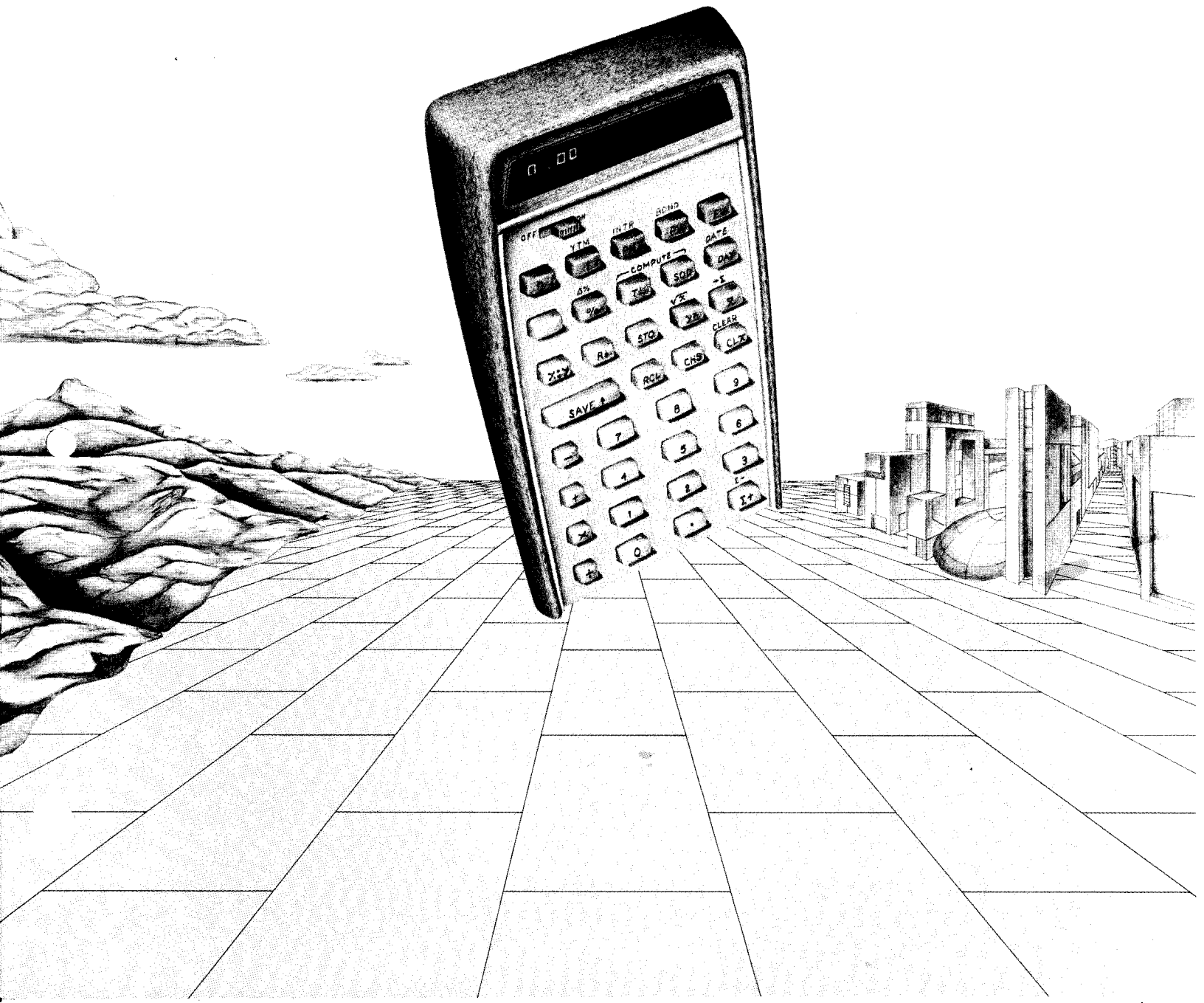
HEWLETT-PACKARD

KEYBOARD

VOL. 5 NO. 1



NEW FIELD EDITORS

It is a pleasure to announce the addition of three new field editors to the *KEYBOARD* staff. Jaroslav Byma of the Hewlett-Packard Intercontinental office in Palo Alto, California will be responsible for all of Asia except Japan. Larry Gillard is the new field editor for Canada, in addition to being the Canadian Sales Manager for HP Calculator Products. Ed Jaramillo is the field editor for Latin America. Ed, located at HP Intercontinental, replaces Daniel Mazar Barnett, who has transferred to the HP sales office in Sao Paulo, Brazil.

Any of the field editors will gladly accept your programs, programming tips, and letters to *KEYBOARD*.

APPLICATIONS INFORMATION
FOR HEWLETT-PACKARD CALCULATORS
PUBLISHED AT P.O. BOX 301, LOVELAND, COLORADO 80537
Editor: A.B.Sperry
Art Director: L.E.Braden
Artist/Illustrator: H.V.Andersen

Field Editors: **ASIA**--Jaroslav Byma, Hewlett-Packard Intercontinental, 3200 Hillview Avenue, Palo Alto, California 94304; **AUSTRALASIA**-Bill Thomas, Hewlett-Packard Australia Pty. Ltd., 22-26 Weir Street, Glen Iris, 3146, Victoria; **CANADA**--Larry Gillard, Hewlett-Packard Canada Ltd., 275 Hymus Blvd., Pointe Claire, Quebec; **EUROPE**-Christian Langfelder, Hewlett-Packard GmbH, Herrenberger Strasse 110, 703 Böblingen , West Germany; **JAPAN**-Akira Saito, Yokogawa-Hewlett-Packard Ltd., 59-1, Yoyogi 1-chrome, Shibuya-ku, Tokyo 151; **LATIN AMERICA**--Ed Jaramillo, Hewlett-Packard Intercontinental, 3200 Hillview Avenue, Palo Alto, California 94304; **SOUTH AFRICA**-Dennis du Buisson, Hewlett-Packard South Africa (Pty.) Ltd., 30 de Beer Street, Braamfontein; **EASTERN U.S.A.**-Stan Kowalewski, Hewlett-Packard Co., W120 Century Road, Paramus, New Jersey 07652; **MIDWESTERN U.S.A.**-Jerry Reinker, Hewlett-Packard Co., 3460 South Dixie Drive, Dayton, Ohio 45439; **SOUTHERN U.S.A.**-Bob McCoy, Hewlett-Packard Co., Post Office Box 28234, Atlanta, Georgia 30328; **WESTERN U.S.A.**--Robert C. Reade, Hewlett-Packard Co., 3939 Lankershim Boulevard, North Hollywood, California 91604.

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

TO HP *KEYBOARD* Readers

The continuing goals of *KEYBOARD* include the publication of articles describing HP calculators and peripherals, software, interesting applications of calculator systems, and useful programming tips and programs for a variety of applications.

While it is possible to cover only a limited number of calculator applications in one *KEYBOARD* issue, we are trying to bring you articles of interest in many fields over a period of time. Your comments and suggestions are always welcome, as well as any programs and programming tips you are willing to share with other users. Programs of sufficient interest which are not published in *KEYBOARD* due to space limitations or other reasons are published in the HP Calculator Program Catalog.

The new BASIC language Model 30 Calculator was described in the last issue of *KEYBOARD*. In contrast, the new HP-80 Financial Calculator featured in this issue can be carried in a man's shirt pocket, but contains built-in functions to solve bond and loan calculations at the touch of a few keys. The postcard in this issue will bring you more information on either calculator.

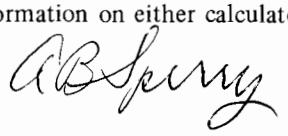


TABLE OF CONTENTS

HP-

THE TIME AND MONEY

MACH E

POWER

Introduced on January 16, 1973, the new pocket-sized HP-80 Calculator is a unique tool for solving virtually all financial calculations involving the relationship between time and money. Present and future values, bond yields, trend lines, and mean and standard deviation are some of the numerous business calculations which the HP-80 performs quickly with a few keystrokes. This nine-ounce (255 g) device has a computer-like solid-state memory, giving it capability far beyond simple addition, subtraction, multiplication, and division. The HP-80 is shown full-size on the facing page.

Although Hewlett-Packard is widely known as a manufacturer of electronic test equipment, in recent years the company has diversified into the computer and calculator fields. About a year ago it introduced its first pocket-sized scientific calculator, the HP-35. Tens of thousands of these units are now in use by engineers and scientists throughout the world.

The HP-80, the powerful business counterpart of the HP-35, is aimed at providing the same problem-solving service for the financial community that the HP-35 supplies to the technical world. Some of the HP-80's keys provide two modes of operation, selected by a gold-colored shift key. This allows the operator to solve all types of problems related to interest rate, discounted notes, and other complex business and statistical problems, using programs built into the calculator's memory.

The HP-80 performs calculations with ten significant digits, giving results with high accuracy. This accuracy produces answers to business problems which are often more accurate than the figures in established financial tables. As well as performing addition, subtraction, multiplication, and division, the HP-80 requires only a minimal number of keystrokes to solve these problems:

1. Constant storage
2. Selective round-off
3. Percentage calculation
4. Percent difference
5. Square root
6. Powers (exponentiation)
7. Running total (summation)
8. Mean (arithmetic average)
9. Standard deviation
10. Number of days between two dates
11. Future date given number of days
12. Future value of an amount compounded
13. Present value of an amount compounded
14. Effective rate of return for compounded amounts
15. Number of periods for an amount compounded
16. Future value of an annuity
17. Present value of an annuity
18. Effective rate of a sinking fund
19. Effective rate of a mortgage
20. Installment of an annuity given future value
21. Installment of an annuity given present value
22. Number of periods for a sinking fund
23. Number of periods for a mortgage
24. Add-on to effective annual rate conversion
25. True equivalent annual rate
26. Linear regression (trend-line) analysis
27. Sum-of-the-years' digits depreciation amortization
28. Rule of 78's finance charge amortization
29. Discounted cash flow analysis
30. Accumulated mortgage interest calculation
31. Remaining principal on a mortgage
32. Accrued interest (360 and 365 day year)
33. Discounted notes (360 and 365 day year)
34. Discounted note yields (360 and 365 day year)
35. Bond price
36. Yield-to-maturity of a bond

INSTANT ANSWERS

The HP-80 gives answers to most business problems in seconds, using a few keys to enter data and actuate its built-in programs. In every case it can save minutes compared to using conventional calculations to solve the same problems. This can amount to hours saved each day for repetitive calculations.

For example, take the problem of finding the true rate of interest on a $2000 loan if it is to be repaid at the rate of $500 per year for 5 years:

CONVENTIONAL METHOD

Use basic formula:

$$\frac{PV}{PMT} = \frac{[1 - (1+i)^{-n}]}{i}$$

where: PV = the principal amount
PMT = installment amount per compounding period
i = interest rate per compounding period
n = the number of compounding periods

Thus:

$$\frac{[1 - (1+i)^{-n}]}{i} = \frac{2000}{500} = 4$$

Next, consult present value of an annuity table to find the percentage value closest to 4 (for the number of periods equalling 5, of course). The table value of 7½% for 5 years is 4.0458849 while the value for 8% is 3.99271004. The *approximate* rate of interest is somewhere in between.

Now for interpolation. Let:

X = amount between low and actual value
.005 (or ½%) = difference between two table values
.0458849 = difference between lower table amount of 4.0458849 and actual of 4
.05317486 = difference between higher and lower table amounts

Then, set up the equation as a proportion:

$$\frac{X}{.005} = \frac{.0458849}{.05317486}$$

Cross Multiply:

$$(.05317486)X = .005 \times .0458849 = .0002294245$$

Divide by .05317486:

$$X = \frac{.0002294245}{.05317486} = .00431452946$$

$$= .43\%$$

Thus, first approximation of true annual rate = lower table rate of 7.5% + .43% = 7.93%.

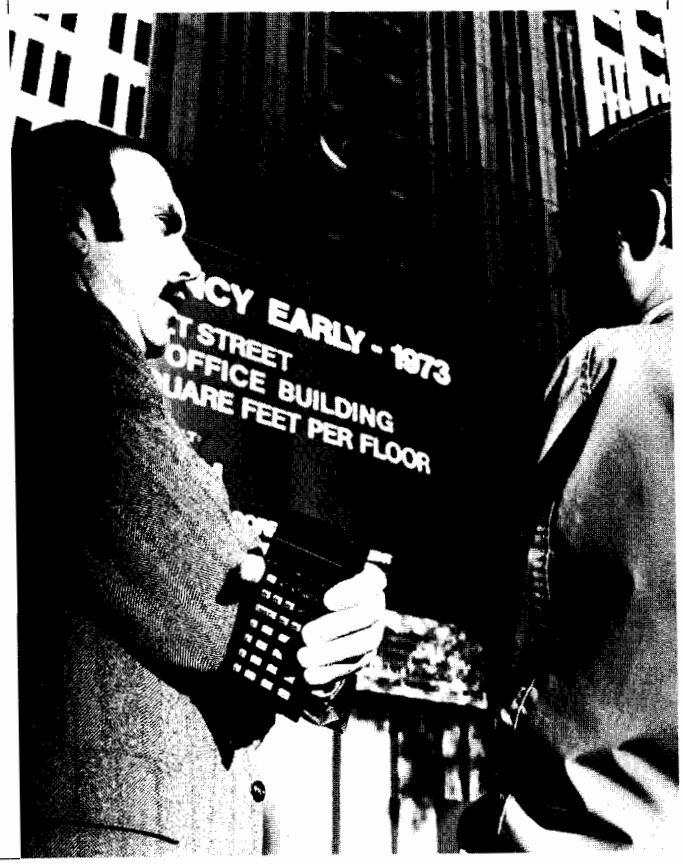
Time elapsed: 6 minutes

HP-80 SOLUTION

5 [n] 500 [PMT] 2000 [PV] [i]

Answer: 7.93 (%)

Time elapsed: 10 seconds

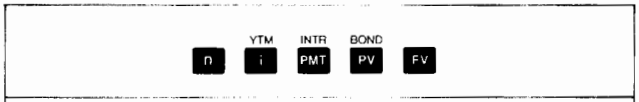


TIME CALCULATIONS

Built into the HP-80's memory is a 200-year calendar, spanning the period from January 1, 1900 through December 31, 2099. This program checks for logic errors, and causes the display to flash when an invalid date is used in a calculation. It also takes into account all leap years. Dates are entered as 7- or 8-digit numbers; for example, April 8, 1978 would be entered as 4.081978.

Possible time calculations include the number of days between two dates; past or future dates, given the starting date and the number of intervening days; and determination of the day of the week corresponding to a particular date.

The mathematical approach used for bond problems with this calendar is more precise than the traditional one established in the 1800's which assumes an arbitrary 30-day month. Although the results from the HP-80 calculation usually differ only beyond the second decimal place from those obtained using tables, the HP-80 owner's manual also gives the traditional calculation method for users wishing their results to conform to the table values.



Enter values left to right, skip key that triggers solution until all values are entered.

n	Number of periods (i.e., days, months, years, etc.)
i	Interest rate per period expressed as a percent (rather than the decimal equivalent)
PMT	Periodic payment or installment portion of an annuity or sinking fund, annual coupon rate on bonds
PV	Present value or principal; that is, the worth right now
FV	Future value after n periods have elapsed
YTM	Yield-to-maturity of a bond
INTR	Interest amount
BOND	Bond price

FINANCIAL FUNCTION KEY SET

There is no need for the user to memorize the various equations used in financial calculations in order to solve the problems on the HP-80. The Quick Reference Guide provides key-by-key instructions for the most commonly used functions, and even that is usually not needed after a few runs through a particular type of problem.

LOGICAL KEYBOARD

The color-coded keys are grouped with related functions near one another for maximum convenience. The black financial keys used for interest calculations in the above example are all in the top row of the keyboard. Other business/extended function keys are in the next two rows. The number keys and arithmetic function keys are grouped together, and the data manipulation keys [x:y], [R↓], [STO], [SAVE↑], [RCL], [CHS], and [CLx] are near the middle of the keyboard for convenience. The gold shift key is near most of the dual-function keys it controls.

TIME-AND-MONEY CALCULATIONS

The sequence of keys used in the sample interest calculation shows entry of the known numbers in a left-to-right sequence on the keyboard. This is a rule for using the financial keys, with either their primary or extended functions. The known values are entered from left to right, skipping the key for the unknown value until the others are entered. Then pressing the key for the unknown triggers the calculation to solve for that value. This sequence is shown below.

ARITHMETIC OPERATION

The HP-80 can be used as an arithmetic four-function (+, −, x, ÷) calculator with simple operating rules. Its functions also include exponentiation (raising a number to a power) using the y^x key, and finding the square root of a number using the same key in the shifted ($\sqrt{x}$) mode.

Four working registers are provided for arithmetic operations. These are arranged in a stack so that entered numbers and intermediate results can be retained to allow chain calculations using any of the arithmetic operators. The value of an expression such as

$$\frac{234 - 5.72}{7.79} \times \frac{3.147 \times 0.426}{4.319 - 3.792}$$

is easily calculated. Simply press keys in the following sequence:

Press	See Displayed
234 [SAVE↑] 5.72 [−]	228.28
7.79 [÷]	29.30
3.147 [×] .426 [×]	39.29
4.319 [SAVE↑] 3.792 [−]	0.53
[÷]	74.55

A brightly-lighted red LED (light-emitting diode) display shows up to ten significant digits. The number appearing in the display can be rounded off to a desired number of digits after the decimal point from 0 to 6. Rounding affects only the display; it does not affect the accuracy of the number internally. Alternately, the display can be set to scientific notation, showing up to 10 significant digits plus a power of 10 indicating the number of places to the right or left of the decimal point. The dynamic range is 10^{-99} to 10^{99}.

The HP-80 operates on either battery power or AC. The battery pack with a normal charge provides three to five hours of operation before recharging. Plugging in the AC adapter/recharger allows operation of the HP-80 during the recharging cycle.

Indicators are provided for improper operation and low battery condition. An erroneous operation such as dividing by zero will cause the display to flash until the operator clears the error. When the battery voltage becomes low, but before it is low enough to cause erratic operation, the decimal point lights up in each position on the display. The calculator can be operated momentarily in this condition before plugging in the recharger or inserting a freshly charged battery pack.

The following accessories are supplied with the purchase of an HP-80 Calculator:

HP-80 Quick Reference Guide
Battery pack
Battery charger/AC adapter
HP-80 Deluxe Travel Safety Case (Holds HP-80 and accessories)
Soft carrying case
HP-80 Owner's Handbook
Personalizing labels (4 each)

Optional accessories which may be purchased separately are an additional battery holder and pack, a security cradle to fasten the calculator to a desk, and a hard leather field case.

Orders for the HP-80 or the HP-35 may be placed or further information obtained by writing directly to

HEWLETT-PACKARD COMPANY
Advanced Products Division
10900 Wolfe Road
Cupertino, California 95014

or by filling out and mailing the postpaid reply card in this *KEYBOARD*.

Heating, Ventilation and Air-conditioning Pac

Program Listing
Part Number 09810-74500

Hewlett-Packard is happy to announce that our Model 10 HVAC Pac is now available. We hope that this set of programs will assist in solving several of the more cumbersome and time-consuming environmental control problems. The packet may be purchased through your local HP sales office. The programs included in the HVAC Pac are described below.

I-1 Design Data Loading

This program stores information concerning design conditions (temperatures, air density ratio, supply air temperature), construction data (U factors, perimeter loss factors) and human heat gain factors for use in Program I-3.

I-2 Solar Heat Gain Data Loading

This program stores on 1 side of a 3-5/8 inch magnetic card 9 solar heat gain factors for glass, 8 total equivalent temperature differentials (TETD) for walls, and 1 TETD for a roof. In addition the date and time of day for which the data are applicable is stored.

I-3 Heat Loss-Heat Gain

This program will compute heat loss alone or with heat gain. Input data is identified and printed. Calculations are performed by room, for any number of rooms. The summary tape, designed to fit on an 11 inch page, includes subtotals of the sources of heat gain as well as the grand total heat loss (GTHL) and grand total heat gain (GTHG). Shr, Tons AC, Sq. Ft./Ton AC, CFM, and CFM/Sq. Ft. given for heat gain.

I-4 Building Peak Solar Heat Gain

This program determines the date and time of the maximum solar heat gain for a building given the total (UXA) for each orientation of the walls of the building and the total (AXSF) for each orientation of the glass of the building. The variable input of this program is the solar heat gain data stored on 3-5/8 inch magnetic cards by Program I-2.

I-5 Heat Gain

This program performs the same function as the heat gain portion of Program I-3. A summary tape is generated for each room. The tape includes the room identification number, the date and time of maximum heat gain, the sources of heat gain as well as the total sensible heat gain, and the grand total heat gain. Also given are the sensible heat ratio, tons of air conditioning, CFM and other information.

II-1 Duct Sizing-Static Regain Method

This program uses the static regain method to size the ductwork for a duct system. Is applicable to any altitude and for any regain factor.

II-2 Water Pipe Sizing

This program will size schedule 40 steel pipe for branches, submains, and mains carrying chilled water. The user may enter a conversion factor which will enable the program to convert tons of air conditioning to gallons per minute of chilled water.

II-3 Gas Pipe Sizing (Pole Formula)

This program will size schedule 40 steel pipe for low pressure gas (specific gravity = 0.60) based on the pole formula (up to 10 inches water column). The program will size the pipe so the pressure drop to each appliance is the same except for differences caused by the use of commercially available steel pipe.

II-4 Piping System Deflection

This program calculates the deflection in the X, Y, and Z directions for a piping system.

III-1 U Factor Calculation

This program provides a quick and simple method for calculating U factors (overall coefficients of heat transmission) from component resistances.

III-2 Temperature in an Unheated Space

This program provides a quick and simple method for calculating the temperature of an unheated space.

III-3 Payroll Program

This is a program to calculate a payroll. The input includes: (1) the month and day corresponding to the payroll period ending date, and (2) the information for each employee--regular hours, overtime hours, and extraordinary pay for the period. The program computes all taxes, deductions, and total pay. It maintains up-to-date totals for each individual, and payroll period totals for the company.

Tape Operating Systems Make Data Storage Easy

As calculator systems increase in power and complexity, greater skill is needed to operate a system to its fullest extent. And as peripherals are added it is not only necessary to learn and understand their individual properties, but also how they can interact. The growth of computer systems created these same problems (but on a larger scale) for their users. These problems were effectively solved by developing *operating systems* to reduce the need for detailed technical understanding of exactly how things happen within the system. The principle of an operating system can also enhance the operation of sophisticated calculator-systems.

WHAT IS AN OPERATING SYSTEM?

An operating system (O/S) is a program that simplifies system operation by making decisions and doing things that would otherwise be tedious, complicated, or too slow if done manually. The inputs to the program are commands to make a decision or do something. An operating system provides the user with straightforward, high-level ways of performing certain activities.

Operating systems simplify program handling. For instance, it is an O/S that enables a computer in a computer center to *automatically* run one program after another (batch processing). It is that O/S that understands the job control cards that preface each deck of punched cards, and that checks the user numbers and bills running time to their accounts. It is the O/S that enables a computer to run two or more programs 'simultaneously.' The operating system establishes the program linkage that permits a main program to call any program in an entire library of subroutines that exists within the system.

Operating systems simplify the handling and storage of large quantities of data, as well as programs. Bulk memory devices, such as magnetic tapes and discs, can store both. But the tape and disc units themselves do not recognize programs as programs, nor data as data; they simply store the sequence of codes supplied to them. However, the O/S for the disc or tape can recognize a program, and where it starts and ends in the memory, and can recognize data, and where it is stored. The O/S can also transfer these things to and from the memory device, and do it singly, or by entire collections that are identified by name.

One way that an operating system can enhance bulk data storage devices is by establishing simplified conventions for communication between the mainframe and the storage device.

For Model 10 or 20 Calculators, such a system based on the 9865A Cassette Memory makes the calculator appear to have an additional thousand or more 'internal registers' for data storage. The behavior of these registers is similar to that of the 'genuine registers.' This additional memory is accessed through commands to a special operating system. The program that is the O/S must be resident in the calculator's memory during these tape-memory operations.

THE MODEL 10 TOS

The Model 10 Tape Operating System (TOS) establishes a tape-memory of 1000 or more registers located in the 9865A Cassette Memory.* The system also provides for easy storage, recall, and linking of up to 20 programs.** Programs are identified by the numbers 1, 2, 3, ..., n.

*The TOS can be modified to provide as many registers as there is room for on the tape.

**Tape not used for data registers is available for program storage.

Data storage and recall are accomplished with the commands shown in Fig. 1 and 2. Registers are identified by an *index* in the X register. Indexing for the tape-registers starts at one and goes as high as determined by adjustable limits within the TOS. If an alpha option block is available, an error message is printed if the index is out of range.

The index that is placed in X can be computed; it is the integer value of the index that is used by the system.

As illustrated by Fig. 1 and 2, keyboard commands begin with GO TO LABEL ---, and terminate with CONTINUE. The CONTINUE does *not* cause the user's programming to begin running; only the command to the O/S is performed. Commands written as steps in a program begin with GO TO SUB LABEL ---. They are treated as ordinary subroutines written for GTO SUB. These things are true for all eleven commands to the TOS.

There are commands for doing arithmetic directly upon the tape-registers. For instance, to divide the contents of the tape-register 250 by a divisor:

1. Put the divisor in Y.
2. Put the index (250) in X.
3. Give the command 'LBL ÷.'

The answer is placed in Y, and also into the tape-register just operated upon (250). X and Z remain unchanged. The complete set of arithmetic commands is:

LBL +
LBL -
LBL x
LBL ÷

Two other useful data operations are provided. LABEL PRINT prints the index and contents of all consecutive tape-registers between any two specified indices. LABEL ROLL-UP stores any specified constant into all consecutive tape-registers between any two given indices.

```
#
     1.
1.0000

#
     2.
2.0000


#
    23.
-1.0000

#
    24.
-1.0000

#
    25.
-1.0000
```

RESULTS OF THE LBL PRT COMMAND

There are three program handling commands. LABEL UP transfers the programming (from step 0000 to the first End) into the program location indexed by X. The remaining two commands bring the program indexed by X back into the calculator, starting at step 0000. LABEL DOWN brings the program into the calculator but does not start it (for editing), while LABEL GO TO brings it in and starts it running (for linking programs).

Display	From Keyboard	As Program Steps	Result
Z ---	GTO	GTO	Data in Y is stored in the tape-register specified by index.
Y data	LBL	SUB	
X index	YTO	LBL	
	CNT	YTO	

Fig. 1 DATA STORAGE INTO THE TAPE-MEMORY

Display	From Keyboard	As Program Steps	Result
Z ---	GTO	GTO	Z ---
Y ---	LBL	SUB	Y ---
X index	XFR	LBL	X data
	CNT	XFR	

Fig. 2 DATA RECALL FROM THE TAPE-MEMORY

THE

The Model 20 Tape Operating System establishes a tape-memory of 2000 or more registers on the cassette. The names of these tape-registers are T1, T2, T3, ..., Tn. By using the User Definable Functions ROM a key whose mnemonic is T is defined. This key is used in nearly the same way as the R( ) key is used to designate R registers. Think of T1 as a register which is similar to R1. Then:

```
T1+T2→A      is like    R1+R2→A
T(A+B)→X     is like    R(A+B)→A
10TX→A       is like    10RX→A

TT10→A       is like    RR10→A
PRT T5       is like    PRT R5
IF T1=T2;    is like    IF R1=R2;
```

R registers and T registers can be intermixed:

```
T1+R1→A
TR5→A
RT5→A
T15R25→A
```

However, storing numbers into the T registers is not done in the same way as with the R registers. Two additional keys are needed, and their mnemonics are PUT and →T. Then:

```
                ↙one key
PUT 15→T99      is like    15→R99
PUT T11→T25     is like    R11→R25
PUT (A+B)→T7    is like    A+B→R7
```

There is another significant difference between the T's and the R's. Programming, and the presence of ROM's, reduce the number of available R registers, but the number of available T registers is determined solely by the TOS.

Another part of the TOS is a callable subroutine named DATA-MAT. Fig. 3 illustrates the operation of DATA-MAT.

DATA-MAT can be called from within a program, as well as from the keyboard. In either instance, if the call is followed by a leading parameter of zero, DATA-MAT will request directions as illustrated by Fig. 3. DATA-MAT can also be told directly what to do by calling it with a coded list of parameters. For instance:

```
                                  specifies constant storage
CLL DATA-MAT 2,0,1,50
```

That line instructs DATA-MAT to store the constant zero into T1 through T50. In this way, supervisory programs can take advantage of the power and convenience of DATA-MAT while establishing or editing collections of data in the T registers.

Program handling is accomplished with the File-by-Name special program that is supplied with each 11223A Cassette Memory ROM. This is a machine language program that enables the standard load-from-file (LDF) and record-into-file (RCF) commands to operate on files designated by alphameric names, e.g.:

```
. . . .;LDF "STAT". . .
. . . .;RCF "LOANS". . .
```

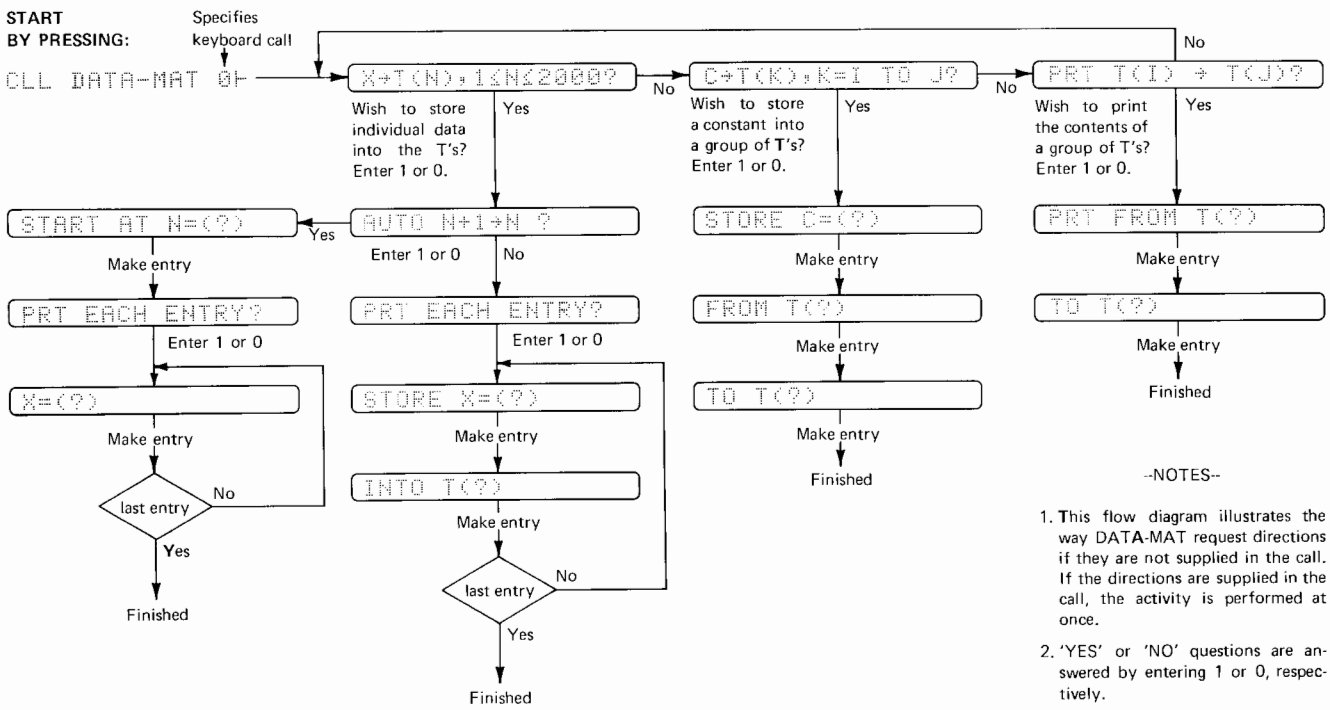


Fig. 3 THE ESSENTIALS OF DATA-MAT

Here are some things to keep in mind while considering a Model 10 or 20 TOS for an application.

The first is this: While they are considerably easier to use than the file-oriented approach used by the Cassette ROM's, each TOS described is both somewhat slower and less efficient than the file-oriented approach. However, these tape operating systems are well-suited for applications where convenience is desired and maximum speed is not essential.

These tape operating systems are neither machine language-special programs nor ROM's; they are written in each calculator's keyboard language, and occupy a substantial amount of memory (see the box at the end of the article). Also, each TOS requires the use of approximately 25 data registers.

Each TOS requires that the Cassette Memory ROM for the calculator be installed. For the Model 20 TOS the UDF ROM is also required.

An operating system reduces the amount of detailed understanding needed to use a system effectively. The tape operating systems described reduce or eliminate the need to learn and implement the file-oriented approach to data storage.

The Model 10 and 20 Tape Operating Systems can make your calculator into at least a 1000 register machine, by establishing a special memory on tape. The results are simplification for user programs employing cassette tapes for data storage, and easier ways to store data onto tape from the keyboard.

The Model 10 and 20 Tape Operating Systems can be purchased through your local sales office. Each TOS is prerecorded to match a typical configuration (easily modified to fit your system), and is supplied with complete operating information. Listed below is some pertinant technical information. BE SURE TO NOTICE WHICH OPTIONS AND ROM'S ARE REQUIRED FOR THE TOS TO OPERATE IN YOUR SYSTEM. Each TOS also requires a 9865A Cassette Memory.

MODEL 10 TOS

Part No.: 09810-76503

Necessary Options: 1012 or 2036 program steps (Opt. 002 or 003), Printer (Opt. 004)

Recommended Options: 111 data registers (Opt. 001)

Necessary ROM's: Cassette Memory ROM (11262A)

ROM's adding useful features to the TOS: Plotter-Alpha ROM (11261A) or Printer-Alpha (11211A), Peripheral Control-Alpha (11266A).

Memory Required: 700 program steps, starting at step 1300 (with Opt. 003) or step 300 (with Opt. 002). Also uses data registers 18 through 40, or with 111 registers (Opt. 001), registers 78 through 100.

Subroutine Nesting: This TOS nests subroutines 3 deep using GTO SUB. Two levels of nesting are always available to the user. All 5 levels are available if levels 3, 4, and 5 do not use the TOS.

MODEL 20 TOS

Part No. 09820-76502

Necessary Options: 429 registers (Opt. 001)

Necessary ROM's: Cassette Memory ROM (11223A) and User Definable Functions ROM (11222A)

Memory Required: For T, PUT, and →T only, 304 R registers remain available. DATA-MAT requires approximately 200 additional registers. (Because of the amount of memory required by DATA-MAT, each Model 20 TOS is supplied in two versions; one with DATA-MAT and one without it.) File-by-name requires an additional 18-plus registers, although its use is also optional. The 15 highest-numbered R registers are used by the keys T, PUT, and →T. DATA-MAT requires the 20 or 32 highest-numbered R registers, depending upon the operation.

9800 SYSTEM APPLICATION CONTEST

There is still time for contestants outside the USA to submit entries in the contest for unusual applications of HP 9800 Calculator systems. The rules are reprinted below for readers who may have missed the announcements in *KEYBOARD* Vol. 4, Nos. 3 and 4.

Two branches of the contest have been held with different time limits to allow participation by users in all countries. A similar prize, a Series 9800 plug-in ROM (Read-Only-Memory) block of the winner's choice will be awarded in each branch. The USA branch deadline was March 15, 1973. The outside-USA branch will run until April 30, 1973. Here are additional rules:

1. Each entry shall be in the form of an article suitable for publication in *KEYBOARD*, and a publication release shall be included.
2. Entries shall be typed double-spaced on paper approximately 8½ by 11 inches (21,6 cm by 27,9 cm).
3. Pertinent photographs, charts, and other illustrations shall be included. Photographs must be good contrast black-and-white prints between 4 by 5 inches (10,1 cm by 12,7 cm) and 8 by 10 inches (20,3 cm by 25,4 cm). The author's photograph and curriculum vitae shall be included.
4. Entries shall be submitted to either a field editor or directly to HP *KEYBOARD*, P.O. Box 301, Loveland, Colorado, 80537, postmarked not later than the deadline date.
5. Entries become the property of Hewlett-Packard and cannot be returned.
6. Winners will be notified in advance of publication. Winning articles will be published in *KEYBOARD* following the contest deadline dates.
7. A proof copy of any article to be published will be submitted to the author for approval prior to publication.
8. Employees of Hewlett-Packard Company, its affiliates and subsidiaries are not eligible to compete.

PART NO.
09810-76003

THREE DIMENSIONAL TRANSFORMATION SUBROUTINES FOR THE MODEL 10

by Robert W. Conley

INTRODUCTION

There is considerable variety in the program organization of 3-D plotting routines, reflecting differences in both programming style and the nature of the information to be plotted. A routine can be written as a complete program, consisting of a main driving routine and utility computation and plotting subroutines, or it may simply consist of a set of user-callable subroutines. In the latter case, the user provides the overall organization and calling sequence in his own driving routines.

The organization into a user-callable subroutine package is the most general. All plotting problems may be solved with such a package, even though some problems may require more user programming. Those computations which must be executed are the transformation of points in three dimensions to points in two dimensions, and the scaling-plotting operations. The Hewlett-Packard Model 10 and the associated x-y plotter have plotting and scaling commands built into the hardware and software. To generate 3-D plots one needs to add a main controlling routine and subroutines to compute the transformation from three dimensions to two dimensions. The Model 10 3-D package described here consists only of the transformation routines. The user must write his own program to call the 3-D subroutines and to execute the Model 10 plotting commands.

The package consists of two subroutines: a coordinate definition subroutine and a transformation subroutine. The user writes the program to call the coordinate subroutine to initialize the necessary parameters, and then repeatedly uses the transformation subroutine to convert the three-dimensional coordinates of the object into two-dimensional coordinates for plotting. The user-written program also must plot the resulting 2-D point.

The package of routines has been written with the objective of having maximum generality in plotting software. The mathematics of the coordinate transformation from three-space to two-space produces a result identical to a visual or photographic image. The correct position, distance, and perspective information are calculated.

TRANSFORMATION MATHEMATICS

To correctly represent a three-dimensional object in two dimensions, the following strategy may be employed as an intuitive guide to the mathematics. Suppose one wanted a two-dimensional (line drawing) projection of a cube exactly as it appears visually. An easy way to obtain the projection would be to view the cube through a glass window and to trace the outline of the cube onto the glass. The outline thus generated would contain all necessary mathematical properties relating to the size, orientation, and perspective of the image. Of course, shadow effects, texture of the material of the cube, and other similar visual effects would be lost. Nevertheless, when constrained to making a line drawing, the image is complete in its information content. The mathematics of this tracing operation is automatically solved by the Model 10 3-D package.

The necessary and sufficient three-space quantities required in the mathematics of the tracing example are the following:

1. The location of the viewer.
2. The location of the object.
3. The location and orientation of the window.

The location of the viewer is the location where the eye or camera would be placed. This location is specified by a point V = (v1, v2, v3) in a three-space coordinate system. (All coordinates in three dimensions refer to the same arbitrary, but fixed, Cartesian coordinate system.)

The location of the object is not rigorously defined because the object will occupy some volume of space. A typical point chosen for this parameter is a point near the center of the object, and is specified by a three-dimensional point C = (c1, c2, c3).

The location of the glass window, also called the plane of projection, is not completely arbitrary. This plane is assumed to be orthogonal to the line segment connecting points C and V. The line segment C to V and the plane of projection intersect at some point between C and V. The location of this intersection is determined by the user and specified by a scalar quantity SIZE. The origin of the two-dimensional Cartesian coordinate system on the plane of projection will lie on the intersection of the plane and the line from C to V, and will be SIZE units from the viewer. SIZE thereby controls the magnitude of the projected image without altering the perspective or orientation. If SIZE is a small number, then the projection plane will be near the viewer and the traced image will be small. As SIZE is increased, the window moves away from the viewer and the tracing becomes larger.

Editor's Note: This complete program is available through the Calculator Program Catalog.

Mathematically, the plane of projection is represented by two quantities, the location of the origin of the coordinate system on the plane $\underline{O}$ = (o1, o2, o3) and the normal vector to the plane N = (n1, n2, n3). These quantities are computed from equations (1) and (2), where ABS(X) is the absolute value, or length, of the three-space vector X = (x1, x2, x3).

(1) $\underline{O}$ = sC + (1-s)V
(2) N = (V-C)/ABS(C-V), where the scalar quantity s = SIZE/ABS(C-V)

In the intuitive example, a point of the tracing on the glass window is the intersection of the plane of the window and the line between the viewer and the point of the object. Enough parameters are now available to compute this point mathematically. The projection of the three-space point T = (t1, t2, t3) is also a three-space point which is the value of the vector function PROJ defined by equation (3), where A*B is the vector inner product of vectors A and B.

(3) PROJ(T) = aV + (1-a)T, where the scalar quantity a = ($\underline{O}$-T)*N/(V-T)*N

It is still necessary to determine the two-dimensional coordinates of PROJ(T) on the plane of projection. This can be accomplished only by introducing a coordinate system onto the plane and then computing the coordinates of PROJ(T) with respect to this coordinate system. A two-dimensional Cartesian coordinate system for the plane is developed as follows. The unit vector in the y direction, Y = (y1, y2, y3) is defined to be the normalized projection of the vector Q = C + (0, 0, 1). The associated unit vector in the x direction, X = (x1, x2, x3) is then given by the vector cross product of Y with N. These computations are summarized in equations (4), where AxB represents the vector cross product of vectors A and B.

(4) Y = (PROJ(Q) - $\underline{O}$)/ABS(PROJ(Q) - $\underline{O}$)
X = YxN

The two-dimensional coordinates of the projected point are then the vector inner products of the point with X and Y. If P3 = PROJ(T), then the two-space point P2 is given by equation (5).

(5) P2 = ((P3 - $\underline{O}$)*X, (P3 - $\underline{O}$)*Y)

A sequence of points in three-space may be plotted by first projecting them onto the plane using equation (3), computing the two-dimensional coordinates of the projected point by equation (5), and finally plotting the two-dimensional point. Care must be taken to insure that the value of SIZE is sufficiently small to allow the important parts of the image to fit within the scaling of the x-y plotter.

PROGRAMM[illegible]ERATIONS

The program for 3-D to 2-D conversion includes two subroutines. The first, labeled C, initializes the coordinate systems. The second, labeled F, performs the transformation calculations. Routine C must be called before any transformations are attempted. These routines use program storage 1168 to 2034, and data storage 081 to 108 as outlined in Figure 1.

This package was designed for use on the Model 10 without need of the plotter ROM. Subroutines C and F use data storage also used by the ROM routines. Consequently, using the 3-D package and the ROM facilities can be dangerous: one of them may destroy values used by the other. When using the 3-D package, the (x,y) points should be scaled by the user to fall within the interval 0 to 9999 in both x and y. The ROM scaling is then not used, and the pen commands are FMT UP and FMT DN; these commands raise and lower the pen, respectively.

108	c3	C	093	y3	Y
107	c2		092	y2	
106	c1		091	y1	
105	v3	V	090	x3	X
104	v2		089	x2	
103	v1		088	x1	
102		Temporary used by routine C	087		Temporary used by routine F
101			086		
100			085		
099	o3	O	084		Temporary used by routines C and F
098	o2		083		
097	o1		082		
096	n3	N	081		Temporary used by routines C and F
095	n2				
094	n1				

Fig. 1 Data Storage Usage of Routines C and F

The complete 3-D package resides on two Model 10 magnetic cards. Included are subroutines C and F plus a program to print an identifying title and to execute the first call to subroutine C. When this program stops, the user may enter his main program and any user generated subroutines.

The two magnetic cards are loaded beginning at program storage location 0000, and the END and CONTINUE buttons are pressed. The title program writes the information of Figure 2. When subroutine C is called the Model 10 prints the following message:

ENTER
POSITION
COORDINATES

(1) VIEWER

X =

Here the routine stops to allow the user to key into the X register the value of c1; then the CONTINUE button is pressed. In a similar manner, c2, c3, o1, o2, o3, and SIZE are entered. A sample printout from the coordinates routine is shown in Figure 3. Whenever it is desired to alter one of the above input values, subroutine C should be recalled. The coordinate systems remain fixed after each call to C until a subsequent call is made.

```
MODEL 10

3D PACKAGE

PROGRAM BY
ROBERT W. CONLEY
MARCH 1972


THE FOLLOWING
DESCRIBES THE 2
USER CALLABLE
ROUTINES-

1. COORDINATES
   (LABELED C)

 THIS ROUTINE IS
 USED TO COMPUTE
 INITIAL VALUES
 FOR THE PLOTTER
 AS THE USER
 ENTERS THE DATA
 DESCRIBING THE
 GEOMETRY OF HIS
 PROBLEM.

2. 3D CONVERSION
   (LABELED F)

 THIS ROUTINE
 CONVERTS THE 3D
 POINT WITH
 COORDINATES IN
 THE X, Y, Z
 REGISTERS TO A
 2D POINT WITH
 THE COORDINATES
 RETURNED IN THE
 X, Y REGISTERS.

COORDINATES IS
NOW CALLED FOR
THE FIRST INPUT.
```

Fig. 2 Main Title

Subroutine C returns control to the main routine which then prints the end title shown in Figure 4. The initialization is now complete. The coordinate transformations are made by entering the x, y, and z three-space coordinates into the X, Y, and Z registers and calling subroutine F with the DEFINABLE F key; the two-space coordinates are returned in the X and Y registers. Although this may be done manually, a stored program is usually written; program storage 0000 to 1167 and data storage 000 to 080 may be used.

Figures 5 and 6 show two views of the same object from two different locations. The object is the surface produced by a function of two independent variables. The 3-D input parameters for these views are given in Figure 7. The function is plotted from 0 to 20 in both x and y. In each of these figures the viewer is so close that some portions are not visible in the window; the plotter does not lose position synchronization when this occurs, however.

```
ENTER
POSITION
COORDINATES

(1) VIEWER
X=
          20.00
Y=
          20.00
Z=
           3.00

(2) OBJECT
X=
           0.00
Y=
           0.00
Z=
           0.00


ENTER SIZE
           5.00
```

Fig 3 Sample Output from Coordinates Subroutine

```
THE USER MAY NOW
PROGRAM THE
MODEL 10 TO CALL
THESE ROUTINES
USING PROGRAM
STORAGE 0000 TO
1167 AND DATA
STORAGE 000 TO
080.
```

Fig. 4 End Title

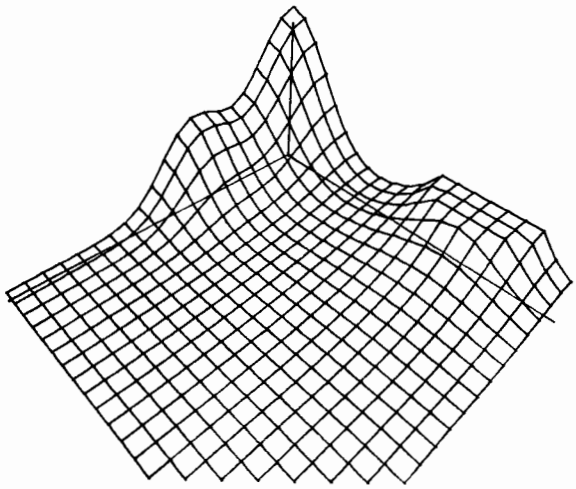


Figure 5

In Figure 5 the viewer is sufficiently distant from the surface that the perspective effects are diminished to a degree where the resulting image approaches an isometric projection. The closer view of Figure 6 demonstrates the perspective capabilities of the transformation package very well.

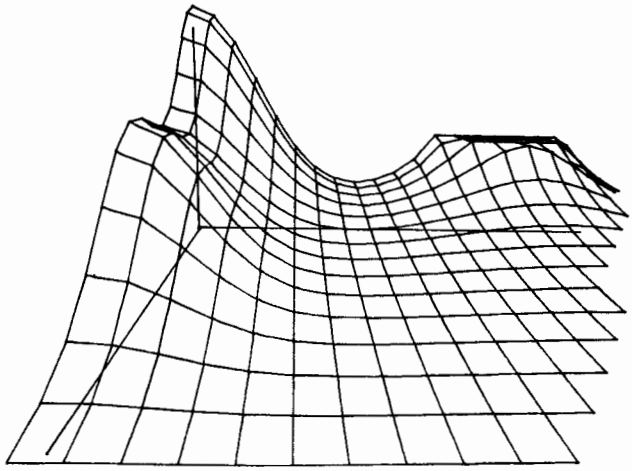


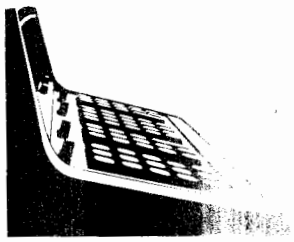
Figure 6

Although the amount of calculation for each transformation is fairly large, the speed of operation is quite acceptable. In the plotting of Figures 5 and 6 the Model 10 could evaluate the successive points in the scanning of the x-y plane, compute the value of z for these points, transform to two dimensions using subroutine F, scale to the plotting field, and plot at rates of approximately two points per second.

Figure Number	Viewer Location	Object Location	SIZE
5	(25,25,25)	(5,5,5)	12
6	(20,5,7.5)	(5,5,5)	8

Figure 7

Robert W. Conley, Jr. is a mathematician at the Air Force Weapons Laboratory, Kirtland Air Force Base, Albuquerque, New Mexico. He received a B.S. in mathematics in 1966 and an M.A. in mathematics in 1968, both from the University of New Mexico. He is currently completing the degree requirements for a Ph.D. in computing science. He was employed in the Research, Biophysics and Analysis Divisions of the Laboratory prior to assignment with the Computational Services Division, Software Section, in September, 1972.



FIRE-DANGER INDEX GENERATION

PART NO. 09100-76018

by Bruce P. McCammon and David A. Rainey

The U.S. Department of Agriculture, Forest Service, maintains the Rocky Mountain Forest and Range Experiment Station at Fort Collins, Colorado in cooperation with Colorado State University. Researchers at the Station have recently developed a National Fire-Danger Rating System (NFDRS). Vast quantities of data have already been accumulated and analyzed. Many computer programs have been developed to aid in this analysis. The program described here is designed to be used with the Hewlett-Packard 9100A-9101A-9120A calculator system. It is a modification of a FORTRAN IV program developed to calculate real time fire danger indices. The HP 9100A version of the program was developed to be used as a research tool and to provide a quick, accurate method of calculating values from the field.

The NFDRS utilizes the fuel model concept. A fuel model is defined to be "a simulated fuel complex for which all the fuel descriptors required for the solution of the mathematical fire spread model (Rothermel, 1972) have been specified" (Deeming et al., 1972). Natural fuels may vary from grass, to brush, to very dense stands of timber, with varying amounts of living and dead fuels present in each. To date, nine fuel models have been defined and are in use by the NFDRS.

The NFDRS considers both dead and living fuels. Living fuels are classified as herbaceous or woody. Dead fuels are classified by their size. Three classes of dead, roundwood fuels are considered:

Fuel Class	Fuel Diameter (inches)
1	<1/4
2	1/4 - 1
3	1 - 3

The fuel complex being evaluated is described by the first seven items input to the program. The data for the nine fuel models may be found in Table 1. All input items are described below.

1. SIGMA1 - The surface-area-to-volume ratio for fuel class 1. This is a measure of the fuel fineness. The units are cm^{-1}.
2. W1, W10, W100, WLIV - These numbers represent the amount of fuels available in each of the fuel classes and the amount of living fuels present. The values are expressed in tons per acre.
3. The slope class code is determined from the actual percent slope of the area being considered:

Slope Class Code	Slope (percent)
1	0 - 20
2	20 - 40
3	>40

4. Bed depth is a measure of the depth (in feet) of the fuels on and adjacent to the ground.
5. Relative humidity is expressed in percent between 0 (zero) and 100.
6. Temperature is entered in degrees Fahrenheit.
7. Windspeed is entered in miles per hour.
8. State of the weather is a code which expresses the amount of cloud cover and kind of precipitation at the time the temperature, humidity, and windspeed are measured.

Code	State of Weather
0	Clear
1	Scattered clouds
2	Broken clouds
3	Overcast
4	Foggy
5	Drizzling
6	Raining
7	Snowing or sleeting
8	Showering
9	Thunderstorm

9. Percent green is a number in the range of 0 (zero) to 50 which represents the amount, by volume, of the herbaceous fuels which are alive. The percent green value integrates the fine, dead fuels with the living herbaceous fuels.

REFER. Deeming, John E., James W. Lancaster, Michael A. Fosberg, R. William Furman, and Mark J. Schroeder. 1972. The National Fire-Danger Rating System. USDA Forest Service Res. Paper RM-84, 165 p. Rocky Mountain Forest and Range Exp. Stn., Fort Collins, Colorado.

Rothermel, Richard C. 1972. A mathematical model for fire spread predictions in wildland fuels. USDA Forest Service Res. Paper INT-115, 40 p. Intermountain Forest and Range Exp. Stn., Ogden, Utah.

Editor's Note: This complete program is available through the Calculator Program Catalog.

The output from the program consists of three danger indices, the Spread Component, the Energy Release Component, and the Burning Index. The Spread Component is a number related to the forward rate of spread of the head of the fire. The Energy Release Component is related to the rate of heat release per unit area within the flaming front at the head of a moving fire. The Burning Index is related to the amount of effort needed to contain a head fire in a particular fuel type within a rating area (Deeming et al., 1972). These indexes are normalized within the program to place them on a 0 (zero) to 100 scale.

Table 1 FUEL MODEL PARAMETERS

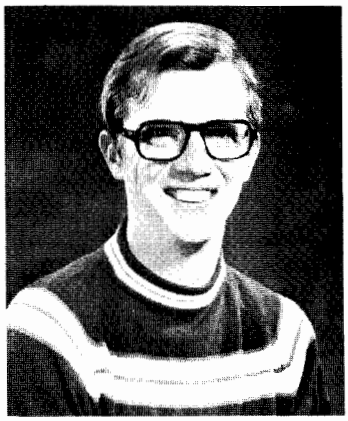
Fuel Model	SIGMA1	W1	W10	W100	WLIV	Bed Depth
A	3000	1.25	0	0	0	.75
B	2000	5.0	4.0	2.0	2.0	6.0
C	2700	1.5	1.0	0	0	1.0
D	1750	1.5	2.5	2.0	0	2.5
E	2500	1.5	1.0	0	0	0.3
F	1500	1.0	0.5	0	2.0	2.0
G	1500	3.0	2.0	5.0	0	1.5
H	2000	1.0	1.0	1.0	0	0.4
I	1500	4.0	5.0	10.0	0	3.5

For this example we will use fuel model C. We will also use a slope class of 1, and a percent green of 20. For the meteorological parameters we will use: temperature = 70, humidity = 40, state of the weather = 3, and wind speed = 10. CONTINUE is pressed after each set of data entries.

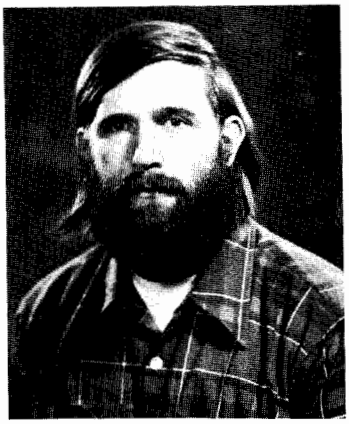
Display	Entry (ies)		
	x	y	z
0 0 1	2700 (SIGMA1)		
0 0 2	1.0 (W10)	1.5 (W1)	
0 0 3	0 (WLIV)	0 (W100)	
0 0 4	1 (bed depth)	1 (slope class code)	
0 0 5	40 (humidity)	70 (temperature)	
0 0 6	10 (windspeed)	3 (state of weather)	
0 0 7	20 (% green)		

At this point the program prints out the three indices as shown below.

4 .. Spread Component
14 .. Energy Release Component
7 .. Burning Index



Mr. McCammon is currently working on a Master's degree at Colorado State University under the sponsorship of the U.S. Forest Service. He received a B.S. degree in Watershed Sciences from Colorado State University in 1971. His hobbies include skiing, hiking, and photography.



Mr. Rainey is a Mathematical Technician working in the National Fire-Danger Rating Project at the Rocky Mountain Forest and Range Experiment Station of the U.S. Forest Service in Fort Collins, Colorado. He attended Colorado State University. Hobbies are backpacking and skiing.

NEW PROGRAM CATALOGS

The 1973 Calculator Program Catalogs are now available. The Series 9100 Catalog lists all programs for the 9100A and 9100B Calculators. The Series 9800 Catalog lists all programs for the 9810A, 9820A, and 9830A Calculators.

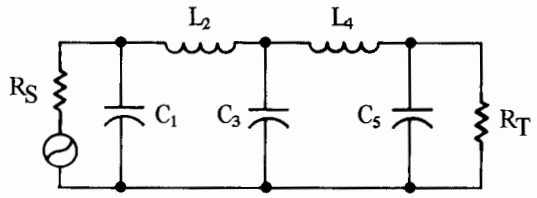
A free copy of either catalog may be obtained by indicating your choice on the enclosed reply card.

Correction

Our thanks go to M. Blamire, Erie Technological Products of Canada, Inc., for the following corrections to the article, "Chebyshev Low-Pass Filter Design, Unequal Terminations," Hewlett-Packard *KEYBOARD*, Vol. 4, No. 3, p. 6-8, 1972.

1) Towards the bottom of page 6,
... inductive input or conductive input filter should read:
... inductive input or capacitive input filter.

2) The schematic at the top right of page 7 should be:



CAPACITIVE INPUT FILTER

3) The example on the right of page 7 should be:

C =
1.20369E - 07
L =
6.40334E - 04
C =
3.15584E - 07
L =
7.80827E - 04
C =
2.35824E - 07

LETTERS TO THE EDITOR

Dear Editor:

Subject: Game of Life

In *KEYBOARD* Vol. 4, No. 3 you gave an article on a Game of Life. I was much interested, then I tried it last August on a Model 20, that Hewlett-Packard of Lyon did send us for a week of evaluation.

The rules were given in the French revue Science Et Vie, July, and attributed to the mathematician Neumann. So I took is as an exercise, and indeed it gave me a good look on the powerful possibilities of that computer.

Without plotter, I could only bring out a field 10 x 10. But then, a matrix is not needed. Each line is stored and handled as a number. For example, to build up the influence field of cells, 10.1 RX is added to 11.1 R(X-1) + 11.1R(X+1). Only the final analysis, to decide survival, birth or death, is made digit by digit, by a single logical stroke. The program stops if no operation (stable figure or empty field).

As I was fully novice in programming the 20, I am aware it can be done in a better way. But it works. Of course, the field is much too small, and can give only an idea of the richness of the game.

Hoping having soon a Model 20 to use, to enjoy working on it, I remain truly yours,

G. Bourquardez
Research Department
Helicopter Division
Aerospatiale, BP 888
13 Marignane, France

```
0:
FXD 0;TBL 4;SPC
;PRT "      NEUMA
NN";SPC 2⊢
1:
ENT "IJ",A;JMP 1
+FLG 13⊢
2:
PRT A;A/10→Z;R(1
+Z)+TN↑ (9-A+10
INT Z)→R(1+Z);
GTO -1⊢
3:
0→X;SPC 2⊢
4:
PRT 1E10/9-R(1+X
+X)→R(13+X);JMP
X=10⊢
5:
SPC ;0→X⊢
6:
10.1R(1+X→X)+11.
1(R(X-1)+R(X+1))
→Z⊢
7:
INT Z-1E10INT (Z
/1E10)→R(X+25);
JMP -1+2(X=10)⊢
8:
0→X⊢
9:
R(1+X→X)/1E10→A;
0→RX→C;R(25+X)/1
E10→R12⊢
10:
10A-(INT (10A)→B
)→A;10R12-(INT (
10R12)→Y)→R12;1+
C→C⊢
11:
JMP 1+((B=0)(Y=3
)+(B=1)((2>Y)+(Y
>3)))⊢
12:
10RX+B→RX;JMP -2
+4(C=10)⊢
13:
10RX+1-B→RX;SFG
10;JMP -3+4(C=10
)⊢
14:
JMP -5+6(X=10)⊢
15:
0→X→C⊢
16:
C+R(1+X→X)→C;
JMP X=10⊢
```

```
17:
JMP 1+(C=0)⊢
18:
IF FLG 10;CFG 10
;0→X;GTO 4⊢
19:
SPC 2;PRT "     T
ERMINE"⊢
20:
END ⊢
```

NEUMANN

```
                43
                44
                45
                46

      1111111111
      1111111111
      1111111111
      1111111111
      1110000111
      1111111111
      1111111111
      1111111111
      1111111111
      1111111111

      1111111111
      1111111111
      1111111111
      1111001111
      1111001111
      1111001111
      1111111111
      1111111111
      1111111111
      1111111111

      1111111111
      1111111111
      1111111111
      1111001111
      1110110111
      1111001111
      1111111111
      1111111111
      1111111111
      1111111111
```

TERMINE

Dear Sir,

Demonstration Stops - 9100A/B

The method of Dr. Butler and Mr. Turner (*KEYBOARD* 4, 2, 36) can be simplified. Program:

IF FLAG
SET FLAG
STOP

Then CONTINUE leaves the flag set, but IF FLAG (manually) and CONTINUE clear it - for the remainder of the program.

Yours faithfully,

P. Wright
British Gas Corporation
Research & Development Division
London Research Station
Michael Road, London SW6 2AD

SILLY MODEL 20 USER

Having unwrapped my Model 20, and feeling after five minutes that I had mastered my machine I decided to insult it by punching in PRINT "YOU ARE A SILLY MACHINE." Guess the reply: YOU ARE A SILLY. Try it, it may do it to you.

Janis Antonovics, B.A., Ph.D.
Department of Botany &
University Program in Genetics
Duke University
Durham, North Carolina

PROGRAMMING TIPS

MODEL 20 INTEGER X WITHOUT MATH ROM

The basic idea for this programming tip was submitted by Prof. Anthony F. Gangi, Texas A&M University. It is an improved version of the technique shown in *KEYBOARD* Vol. 4, No. 2 for obtaining INTEGER X using the 9820A without a Math ROM. It has the advantage of working with negative numbers and numbers in the range 0 to ±1.5, which were not usable with the originally published technique.

Line 3 in the program shown here contains the main INTEGER X routine. Lines 1 and 2 take into account the special case where X = ±1.4999999999. In this case line 3 would otherwise fail, since the Model 20 would round this to ±1.5 for display, printing, and comparison purposes, but retain the exact input number for calculations. X would then be reduced to ±0.999999999, resulting in INT X = 0. Lines 1 and 2 circumvent this and make the function continuous for all positive and negative numbers where $|X| \leq 10^{10}-1$.

INTEGER X PROGRAM

```
0:
ENT "NONINT X =
?",X;PRT "NONINT
 X =";PRT X;SPC
1:
IF X=1.5;1→X
2:
IF X=-1.5;-1→X
3:
PRT "INT X =",X-
.5((1.5≤X)-(X≤-1
.5))+1E11-1E11;
SPC 3
4:
GTO 0
5:
END
```

REDUCING FORWARD SEARCH TIME IN CASSETTE APPLICATIONS

The method presented here is useful in any application requiring the use of large files (sizes greater than 50 registers) on the Cassette Memory. A forward search is initiated by the LOAD FILE, FIND FILE, or the RECORD INTO FILE commands when the current file number is less than the file number being searched for. The forward search procedure is to fast search up to the file before the one in question and then do a slow search until the file in question is found. A significant delay occurs when the slow search is done through a long file. This delay does not occur in a backward search because backward searches are performed entirely in the fast mode.

When the tape is initially marked, insert a short file (minimum 1 register) between any two files in which the first file is longer than approximately 50 registers. The effect of this added file is that the slow search through a short file is barely detectable. The cost of this reduced search time is 54 words for header information for the file and 6 words to store 1 register of data, or 60 words per 1-register file. Since the approximate total number of words on a tape is 44,000, one 1-register file will occupy only .1363% of the total tape capacity. Seventy-three such buffer files will occupy 9.95% of the tape capacity.

CASSETTE COMMANDS

	9810A	9820A
Find File	FMT, 5, 5, CLX	FDF
Load Program	FMT, 5, 5, CNT (S/R)	LDF
Load Data	FMT, 5, 5, XFR	LDF
Record Program	FMT, 5, 5, K	RCF
Record Data	FMT, 5, 5, XTO	RCF

	9830A
Find File	FIND
Load Program	LOAD
Load Data	LOAD DATA
Record Program	STORE
Record Data	STORE DATA