# Instructional Aids for Teaching How to Use the TI-59 Programmable Calculator

by

Ralph E. Hepp

Working Paper No. 10      1983

# MSU INTERNATIONAL DEVELOPMENT PAPERS

Eric W. Crawford, Carl K. Eicher, and Carl Liedholm,
Co-Editors

The MSU International Development Paper series is designed to further the comparative analysis of international development activities in Africa, Latin America, Asia, and the Near East. The papers report research findings on historical, as well as contemporary, international development problems. The series includes papers on a wide range of topics, such as alternative rural development strategies; nonfarm employment and small-scale industry; housing and construction; farming and marketing systems; food and nutrition policy analysis; economics of rice production in West Africa; technological change, employment, and income distribution; computer techniques for farm and marketing surveys; and farming systems research.

The papers are aimed at teachers, researchers, policy makers, donor agencies, and international development practitioners. Selected papers will be translated into French, Spanish, or Arabic.

Individuals and institutions in Third World countries may receive single copies free of charge. See inside back cover for a list of available papers and their prices. For more information, write to:

MSU International Development Papers
Department of Agricultural Economics
Agriculture Hall
Michigan State University
East Lansing, Michigan  48824-1039
U.S.A.

# INSTRUCTIONAL AIDS FOR TEACHING HOW TO USE THE TI-59 PROGRAMMABLE CALCULATOR

By

Ralph E. Hepp

Department of Agricultural Economics
Michigan State University

1983

*This paper is published by the Department of Agricultural Economics, Michigan State University, under the "Alternative Rural Development Strategies" Cooperative Agreement DAN-1190-A-00-2069-00, U.S. Agency for International Development, Office of Multi-Sectoral Development, Bureau of Science and Technology, Washington, D.C.

# PREFACE

There is a worldwide revolution in small computer technology underway and scientists are struggling to find ways to utilize this new technology to help solve development problems in the Third World. We are pleased to announce a number of papers on microcomputers in international agriculture will be published in our International Development Papers series. The aim of these papers is to provide timely information about the rapidly changing state of the new micro-processing technology and its use in research. The papers are also intended as guides to agricultural and social scientists on choosing, installing, and maintaining microcomputer hardware and software systems in developing countries.

Some of the papers will also document field experiences of selected established projects using new data processing hardware and software. Other papers will concentrate on developing guidelines for establishing and maintaining successful microcomputer and/or programmable calculator installations for agricultural research in developing countries.

The present paper is the fourth of these new papers. It is based on staff work by faculty members and graduate students of the Department of Agricultural Economics, Michigan State University, on cost-effective data collection, management, and analysis techniques for developing country applications. This activity is carried out under the terms of reference of the Alternative Rural Development Strategies Cooperative Agreement--DAN-1190-A-00-2069-00--between the Office of Multi-Sectoral Development, Bureau of Science and Technology of the United States Agency for International Development and the Department of Agricultural Economics at Michigan State University.

Readers are encouraged to submit comments about these new papers on microcomputers and to inform us of their activities in this area. Write directly to: Dr. Michael T. Weber, Acting Director, Alternative Rural Development Strategies Cooperative Agreement, Department of Agricultural Economics, Michigan State University, East Lansing, Michigan 48824-1039.

Eric W. Crawford, Carl K. Eicher, and Carl Liedholm
Co-Editors
MSU International Development Papers

# TABLE OF CONTENTS

REFERENCES

--Personal Programming, A Complete Owner's Manual for TI Programmable 58/59, by the staff of the Texas Instruments Learning Center, Texas Instruments, Incorporated.

--Programmable Calculators, Business Applications, by Julius S. Aronofsky, Robert J. Frame, and Elbert B. Greynolds, Jr., McGraw-Hill Book Company.

# INTRODUCTION

Programmable calculators are making mathematical calculations simpler and more accurate for agricultural producers, agribusiness representatives, extension advisors and researchers. Since the introduction of the Texas Instruments 59 calculator, numerous routines and programs have been developed.

The instructional materials on the operation and programming of the TI/59 are an outcome from teaching students, farmers, research and extension workers on how to use and program the calculator. Numerous visual aids are included for easy reproduction as transparencies to assist instructors. The visual aid number corresponds to a text explanation and background information section which is helpful in explaining the concepts covered on the visual aids.

The visual aids and text are organized into eight lesson units by topic. Objectives for the lesson are stated at the outset of each unit. An instructor who is teaching others in the operation and programming of the TI/59 should supplement the enclosed teaching aids with other reference books and personal experiences in using and programming the TI/59.

Material for the visual aids and text has been taken from the Personal Programming and Business Application books listed in the reference section. Special thanks is extended to Michael Beauregard, Forestry Department, Michigan State University who let me use his visual aids on the TI/59 while developing the materials.

## LESSON 1 TODAY'S ELECTRONIC CALCULATOR OR HANDHELD COMPUTER

Objectives

-- Develop an understanding of programmable calculators available for use in research, extension and business.

-- Identify the advantages and limitations of programmable calculators

### 1-1 BASIC CALCULATOR FUNCTIONS

The programmable calculator has the same features as a basic four-function calculator (designed only to add, subtract, multiply and divide) with the additional feature of a memory register which can store and recall instructions and data. Instructions and data can be stored on magnetic cards and reused in the future without keying them back to memory, and with the print cradle, a printout of data, instruction steps and output can be made available. These abilities transform the basic calculator into a computer, one with a limited storage capacity, but capable of performing the minimum computer functions of reading in both data and instruction, storing data and instructions in memory, performing calculations in the manner prescribed by the instruction, reading or writing out the results and controlling all aspects involved in getting an answer.

## 1-2 ADVANTAGES OF THE PROGRAMMABLE CALCULATOR

1. Economy -- The most advanced Texas Instrument programmable calculator with printer costs less than $400.

2. Portability -- The programmable calculator can literally be carried in a pocket.

3. Speed -- Calculations take seconds. Manual calculations can often involve hours, sometimes days.

4. Accuracy -- Just push the right keys.

5. Problem Solving -- For complicated programs requiring many steps and data entries, a larger computer is necessary. However, for many decision making problems, the programmable calculator is the cheapest, fastest and most accessible.

6. Comprehension -- Almost anyone can use a programmable calculator. Computer science training is not required.

## 1-3 LIMITATIONS OF THE PROGRAMMABLE CALCULATOR

1. Memory -- Limited memory space for programming steps and data storage.

2. Printed output -- Alpha printed output is confined to a few key words.

3. Data Entry -- Not suited for problems with large data handling requirements.

4. Programming Language -- Restricted to the keyboard language designed for the calculator.

5. Task Applications -- Limited to problems dealing with mathematical expressions -- Does not have the versatility and flexibility for task application of a micro-computer.

## 1-4 EXAMPLES OF PROGRAMMABLE CALCULATORS/HANDHELD COMPUTERS

TRS-80 PC-1 and PC-2 -- Programmable in Basic

TI/59 -- Keyboard language

## 1-5 FEATURES OF THE TI/59

Power Adapter: The TI/59 comes with a rechargeable power pack built into the calculator. To recharge the calculator, plug the adapter into a household outlet, with the cord inserted in the right-hand side of the programmable calculator. The power pack will also recharge when it is being used on the printer.

Off-On Switch: The switch is located at the top of the calculator. Push to the right to turn on, left to turn off.

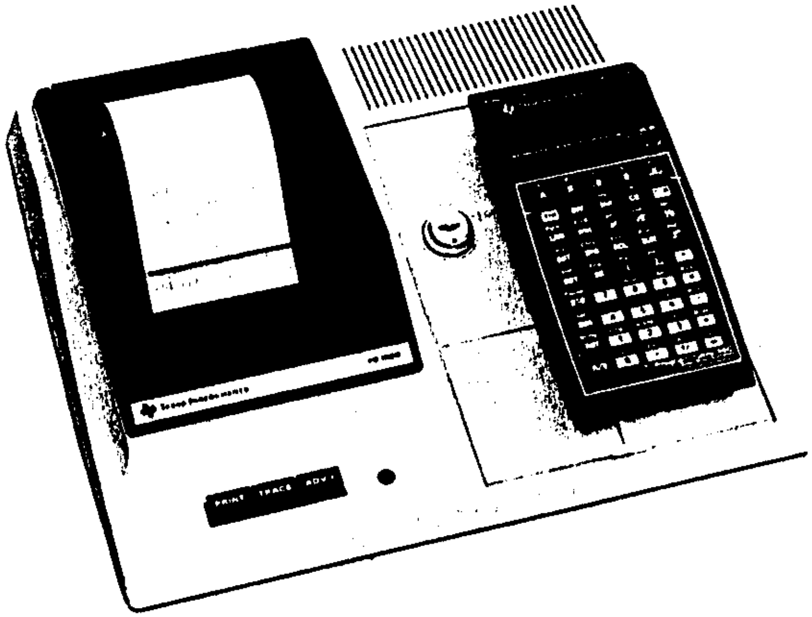
Memory: The TI/59 has 120 memory registers. Each individual memory register can store 8 program steps (or a maximum of 960 program steps) or one number (up to 100 data registers), but not both in one register.

Keyboard: The TI/59 has 45 face keys with the function printed on the key. There are 41 secondary keys with the function printed directly above the key. Face keys are pushed directly, while secondary keys must be preceded by the face key 2nd . This key is similar in function to the shift key on a typewriter which is used to obtain the higher case letters.

Magnetic Cards: Thin plastic strips are used to record the program steps and/or data. A maximum of two cards (480 program steps) can be recorded in the memory registers at one time. Cards are inserted in the right-hand side. The motor in the calculator will pull the cards through. When the buzzer stops, pull the card out.

Printer (PC-100A Print/Security Cradle): The TI/58 and TI/59 can be used with the printer to provide printouts of data, program steps or output. This capability is useful in searching for problems in the program -- "debugging."

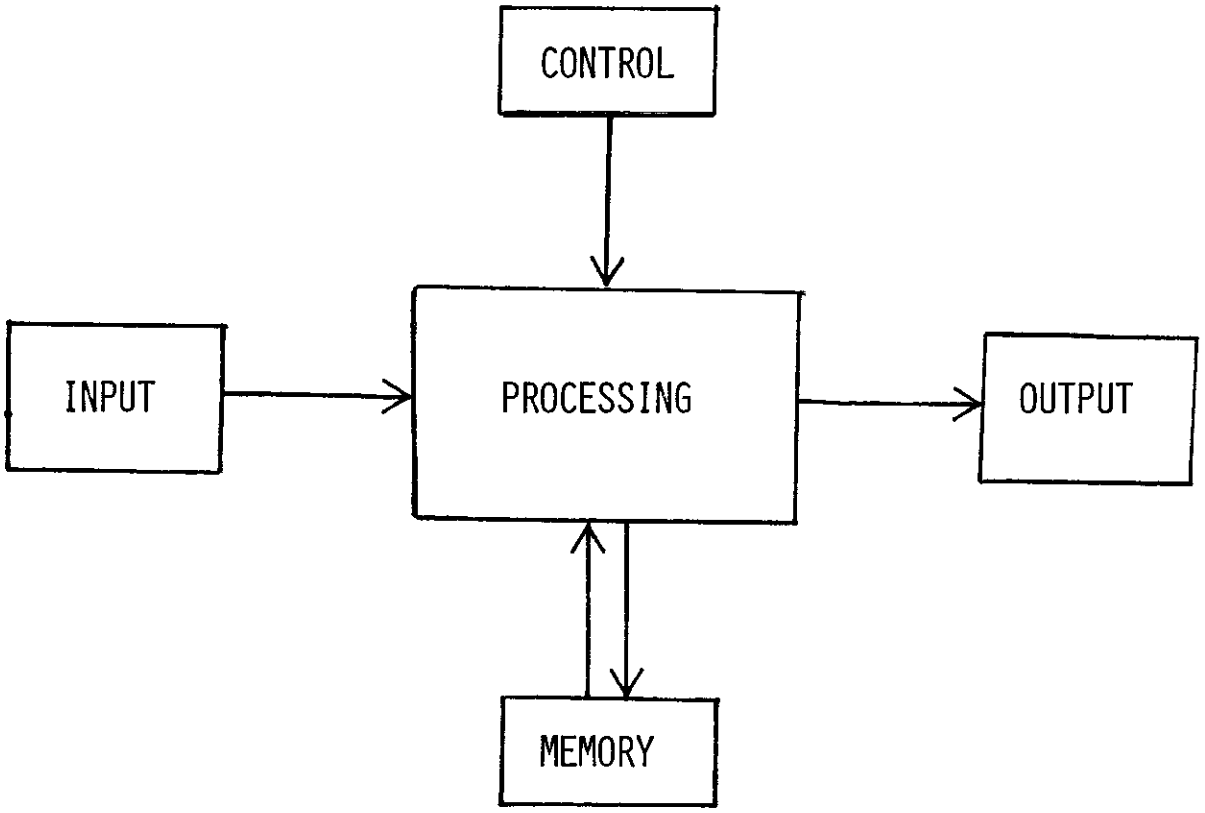
# UNDERSTANDING YOUR TI/59
# PROGRAMMABLE CALCULATOR



DEPARTMENT OF AGRICULTURAL
ECONOMICS

MICHIGAN STATE
UNIVERSITY

BASIC CALCULATOR FUNCTIONS



# ADVANTAGES OF THE PROGRAMMABLE CALCULATOR

1. ECONOMY
2. PORTABILITY
3. SPEED
4. ACCURACY
5. PROBLEM SOLVING
6. COMPREHENSION

# LIMITATIONS OF THE PROGRAMMABLE CALCULATOR

1. MEMORY

2. PRINTED OUTPUT

3. DATA ENTRY

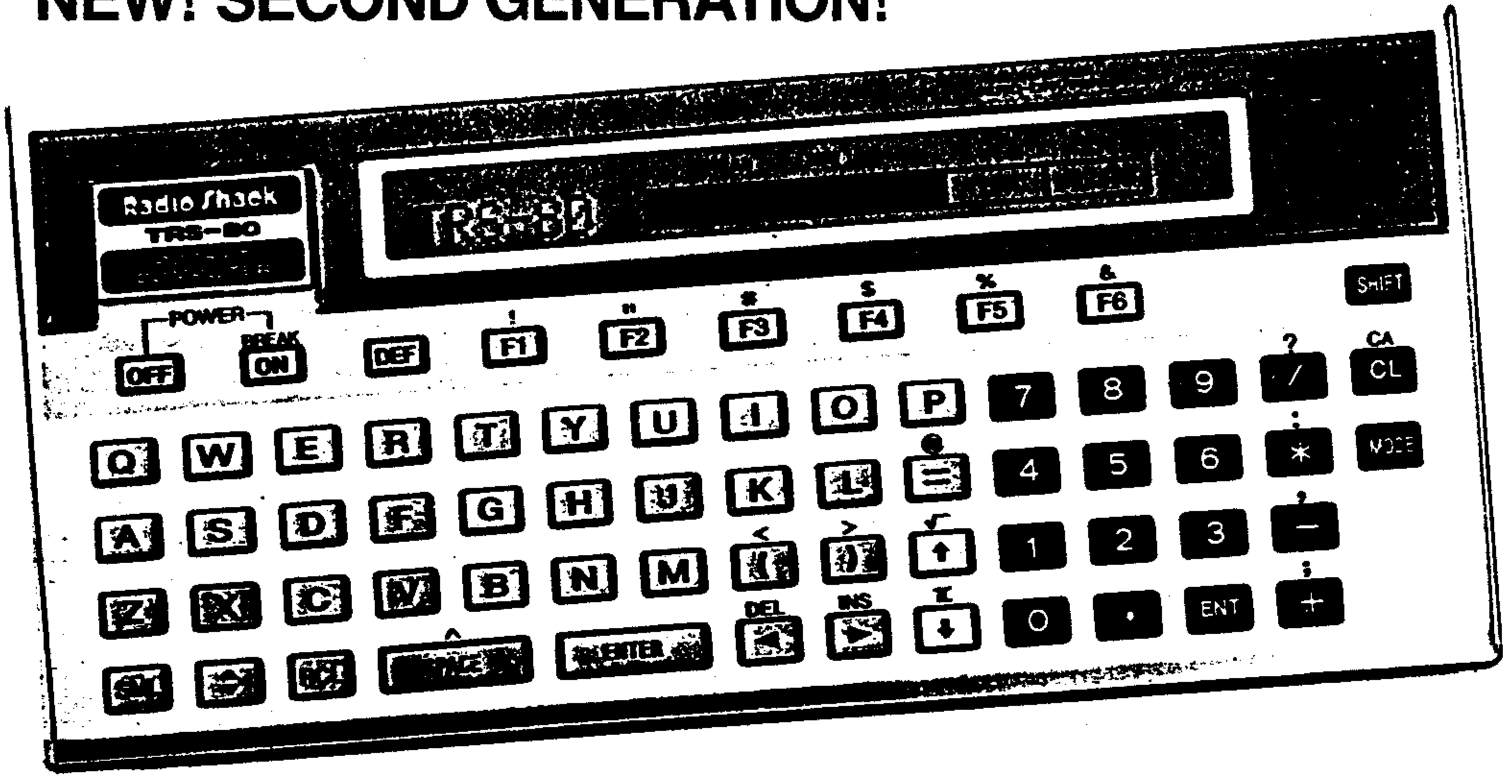
4. PROGRAMMING LANGUAGE

5. TASK APPLICATION

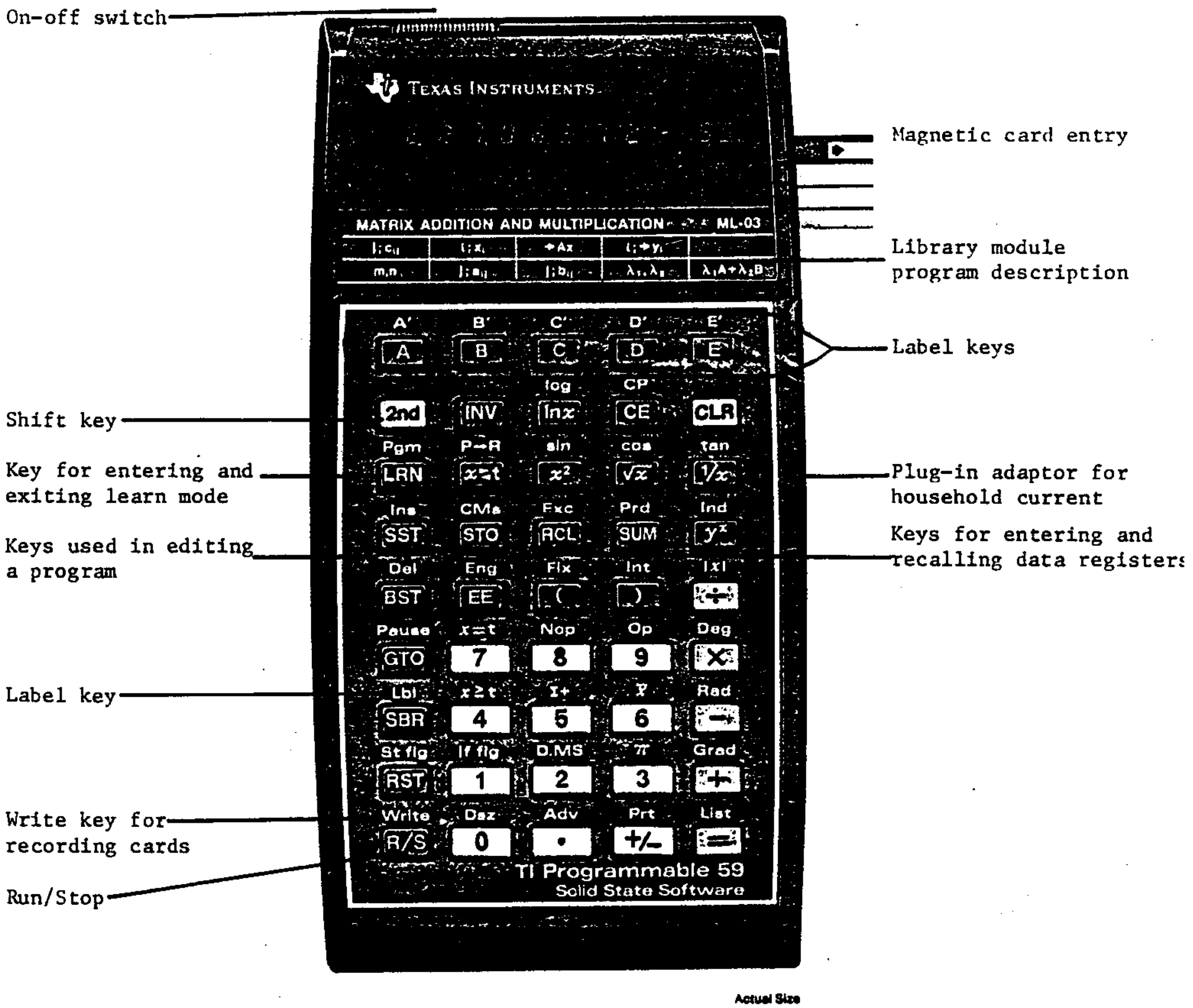
# EXAMPLES OF PROGRAMMABLE CALCULATORS/HANDHELD COMPUTERS

## TRS-80 POCKET COMPUTER MODEL PC-2

## NEW! SECOND GENERATION!



# FEATURES OF THE TI/59


On-off switch
Shift key
Key for entering and exiting learn mode
Keys used in editing a program
Label key
Write key for recording cards
Run/Stop
Magnetic card entry
Library module program description
Label keys
Plug-in adaptor for household current
Keys for entering and recalling data registers
TEXAS INSTRUMENTS
MATRIX ADDITION AND MULTIPLICATION ML-03
TI Programmable 59
Solid State Software
Actual Size

## LESSON 2 MANUAL OPERATIONS

Objectives

-- To become familiar with the features and functions of the keys.

-- To perform arithmetic calculations.

-- To use basic function keys in manual calculations.

-- To be able to store and recall data in calculator memory.

## 2-1 KEYBOARD BASICS

### Clearing the Display

CE -- Clear entry

CLR -- Clear all calculations in progress

### Data Entry Keys

0 – 9 -- Numbers

• -- Floating decimal point

+/- -- Changes the sign of display number

* π -- Places π into the display

## Basic Operation Keys

$\boxed{+}$, $\boxed{-}$, $\boxed{\times}$, $\boxed{\div}$, $\boxed{=}$

## The AOS (Algebraic Operating System) Entry Method

1. Special single variable function Keys -- acts on the displayed number immediately as the function key is pressed -- i.e.: trig and log functions,

2. Powers and roots are handled next,

3. Multiplication and division are completed,

4. Additions and subtractions are completed, and

5. Equals key completes all pending operations.

## Parentheses Keys

$\boxed{(}$, $\boxed{)}$ -- Used to cluster numbers and operations to change the AOS hierachy with the intermost parentheses operation completed first.

## Dual Function Keys

$\boxed{\text{2nd}}$ -- The 2nd Key must preceed the second function key (written above the key)

*Denotes a second key

## 2-2 INVERSE KEY FUNCTIONS

$\boxed{\text{Inv}}$ -- The inverse key increases the number of functions when it precedes the function key

| Function | Inverse Function |
|---|---|
| EE | removes EE |
| ENG | removes ENG |
| Fix | removes Fix |
| log | $10^x$ |
| ln x | $e^x$ |
| $y^x$ | $\sqrt[x]{y}$ |
| Int | fractional part |
| sin | $\sin^{-1}$ |
| cos | $\cos^{-1}$ |
| tan | $\tan^{-1}$ |
| Prod | divide into memory |
| SUM | subtract from memory |
| D.MS | decimal to D.MS |
| P→R | R→P. |
| $\Sigma+$ | $\Sigma-$ |
| $\bar{x}$ | standard deviation |
| list | list data registers |
| SBR | return |
| $x = t$ | $x \neq t$ |
| $x \geq t$ | $x < t$ |
| if flg | if no flag |
| st flg | reset flag |
| Dsz | skip on nonzero |
| Write | read |

## 2-3 MEMORY KEYS

*

$\boxed{\text{CMs}}$ -- Clears all data registers simultaneously.

$\boxed{\text{STO}}$ XX -- Stores the number in the display into data register XX (A two digit data register must be pressed; 00 -- 99)

$\boxed{\text{RCL}}$ XX -- Recalls the contents of data register XX to the display

*

$\boxed{\text{Exc}}$ XX -- Exchanges the contents in data register XX with the number in the display.

## 2-4 MEMORY ARITHMETIC KEYS

$\boxed{\text{SUM}}$ XX -- Sums the contents in the display to the value in data register XX

$\boxed{\text{INV}}$ $\boxed{\text{SUM}}$ XX -- Subtracts the contents in the display to the value in data register XX

*Prd XX -- Multiples the contents in the display by the value in data register XX

INV *Prd XX -- Divides the value in data register XX by the contents in the display

## 2-5 DISPLAY CONTROL -- STANDARD DISPLAY

Display -- 10 digits shown in the display window

Display Register -- Internal register that retains results to 13 digits

Display Form

1. Standard number -- 271124

   or

2. Scientific notation -- 800,000,000,000 is shown as 8. 11 or 8 x $10^{11}$

## 2-6 SCIENTIFIC NOTATION KEY -- EE

Form -- Number = Mantissa x $10^{\text{exponent}}$

Key Sequence

1. Enter the mantissa up to 8 digits (press +/- if negative)

2. Press EE -- Sets up the scientific notation format

3. Enter the exponent up to 2 digits (press $\boxed{\text{+/-}}$ if negative

4. Press any operation key

## 2-7 ENGINEERING NOTATION KEY -- $\overset{*}{\boxed{\text{Eng}}}$

1. Engineering notation is a modified form of scientific notation.

2. The exponent is always adjusted to a multiple of three (ie, $10^3$, $10^6$, $10^{-3}$).

3. The display is converted to engineering notation by pressing $\overset{*}{\boxed{\text{Eng}}}$ .

4. The display returns to the standard format by pressing $\boxed{\text{INV}}$ $\overset{*}{\boxed{\text{Eng}}}$ .

5. A number cannot be entered in engineering notation like scientific notation -- the number must be converted to engineering notation.

## 2-8 FIX-DECIMAL CONTROL -- $\overset{*}{\boxed{\text{Fix}}}$

1. Controls the display of digits to the right of the decimal point.

2. Key sequence -- $\overset{*}{\boxed{\text{Fix}}}$ X where X is the number of digits to the right of the decimal point in the display. X can be from 0-8.

3. The display register retains the internal accuracy to 13 digits.

4. Pressing $\boxed{\text{INV}}$ $\overset{*}{\boxed{\text{Fix}}}$ removes the fix-decimal.

## 2-9 SQUARE, SQUARE ROOT, RECIPROCAL KEYS

Key Sequence

1. Enter a number in display

2. Enter the algebraic function

## 2-10 POWERS AND ROOTS

Key Sequence for Powers $y^x$

1. Enter the number y you want raised

2. Press $\boxed{y^x}$

3. Enter the power x

4. Press $\boxed{=}$ (or any operation key)

## 2-11 OTHER ALGEBRAIC FUNCTIONS

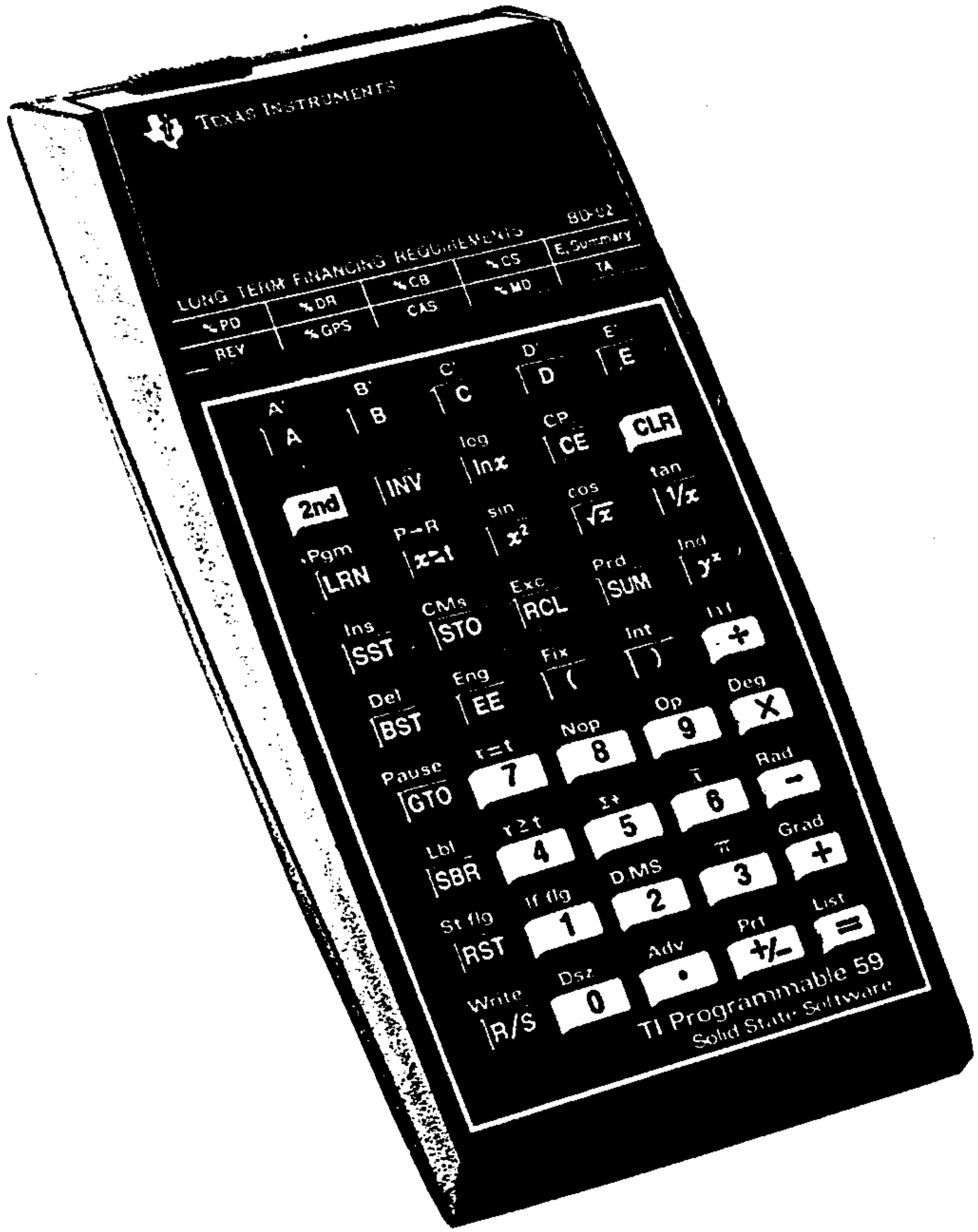
Logarithm -- $\boxed{\text{lnx}}$, $\overset{*}{\boxed{\text{log}}}$

Angular Mode Keys -- $\overset{*}{\boxed{\text{Deg}}}$, $\overset{*}{\boxed{\text{Rad}}}$, $\overset{*}{\boxed{\text{Grad}}}$

Trigonometic Keys -- $\overset{*}{\boxed{\text{sin}}}$, $\overset{*}{\boxed{\text{cos}}}$, $\overset{*}{\boxed{\text{tan}}}$

See p. II-13 & II-14 of Personal Programming for explanation

# TI/59 KEYBOARD BASICS



# INVERSE KEY FUNCTIONS

| Function | Inverse Function |
|---|---|
| EE | removes EE |
| ENG | removes ENG |
| Fix | removes Fix |
| log | $10^x$ |
| ln x | $e^x$ |
| $y^x$ | $\sqrt[x]{y}$ |
| Int | fractional parl |
| sin | $\sin^{-1}$ |
| cos | $\cos^{-1}$ |
| tan | $\tan^{-1}$ |
| Prod | divide into memory |
| SUM | subtract from memory |
| D.MS | decimal to D.MS |
| P→R | R→P. |
| Σ+ | Σ− |
| $\bar{x}$ | standard deviation |
| list | list data registers |
| SBR | return |
| $x = t$ | $x \neq t$ |
| $x \geqslant t$ | $x < t$ |
| if flg | if no flag |
| st flg | reset flag |
| Dsz | skip on nonzero |
| Write | read |

# MEMORY KEYS

*$\boxed{\text{CMs}}$ -- CLEAR MEMORY

$\boxed{\text{STO}}$ xx -- STORE

$\boxed{\text{RCL}}$ xx -- RECALL

*$\boxed{\text{Exc}}$ xx -- MEMORY EXCHANGE

## EXAMPLE

| PRESS | DISPLAY |
|---|---|
| 3.21 $\boxed{\text{STO}}$ 08 | 3.21 |
| $\boxed{\text{CLR}}$ | 0 |
| $\boxed{\text{RCL}}$ 08 | 3.21 |
| 98 | 98 |
| *$\boxed{\text{Exc}}$ 08 | 3.21 |

# MEMORY ARITHMETIC KEYS

$\boxed{\text{SUM}}$ xx -- MEMORY SUM

$\boxed{\text{INV}}$ $\boxed{\text{SUM}}$ xx

*$\boxed{\text{PRD}}$ xx -- MEMORY PRODUCT

$\boxed{\text{INV}}$ *$\boxed{\text{PRD}}$ xx

## EXAMPLE

| PRESS | DISPLAY |
|---|---|
| 25 $\boxed{\text{STO}}$ 10 | 25 |
| 30.5 $\boxed{\text{SUM}}$ 10 | 30.5 |
| $\boxed{\text{RCL}}$ 10 | 55.5 |
| 2 *$\boxed{\text{PRD}}$ 10 | 2 |
| $\boxed{\text{RCL}}$ 10 | 111 |

## STANDARD DISPLAY

### EXAMPLE

| PRESS | DISPLAY |
|---|---|
| 842 X 322 = | 271124 |
| 400000 X | 400000 |
| 2000000 = | 8.11 |

SCIENTIFIC NOTATION KEY -- $\boxed{\text{EE}}$

FORM -- NUMBER = MANTISSA X $10^{\text{EXPONENT}}$

KEY SEQUENCE FOR ENTRY -- EXAMPLE

| PRESS | DISPLAY |
|---|---|
| 3.8901448 | 3.8901448 |
| $\boxed{+/-}$ | -3.8901448 |
| $\boxed{\text{EE}}$ | -3.8901448 00 |
| 32 | -3.8901448 32 |
| $\boxed{+/-}$ | -3.8901448 -32 |

## ENGINEERING NOTATION KEY -- *$\boxed{\text{ENG}}$

EXAMPLE

| PRESS | DISPLAY |
|---|---|
| $\boxed{\text{CLR}}$ *$\boxed{\text{ENG}}$ | 0.00 |
| 8 $\boxed{\text{X}}$ 98 $\boxed{\text{X}}$ | 784.00 |
| 30 $\boxed{=}$ | 23.52 03 |
| $\boxed{\text{INV}}$ *$\boxed{\text{ENG}}$ | 23520 |

## FIX - DECIMAL CONTROL -- $^{*}\boxed{\text{FIX}}$

KEY SEQUENCE -- $^{*}\boxed{\text{FIX}}$ x

EXAMPLE

| PRESS | DISPLAY |
|---|---|
| $\boxed{\text{CLR}}$ | 0. |
| 2 $\boxed{\div}$ 3 $\boxed{=}$ | .6666666667 |
| $^{*}\boxed{\text{FIX}}$ 6 | 0.666667 |
| $^{*}\boxed{\text{FIX}}$ 2 | 0.67 |
| $^{*}\boxed{\text{FIX}}$ 0 | 1. |
| $\boxed{\text{INV}}$ $^{*}\boxed{\text{FIX}}$ | .6666666667 |

## SQUARE, SQUARE ROOT, RECIPROCAL KEYS

EXAMPLE: $\sqrt{4}\ \left(\frac{1}{5}\right)^2 = 50$

| PRESS | DISPLAY |
|---|---|
| $\boxed{\text{CLR}}$ | 0 |
| 4 $\boxed{\sqrt{x}}$ | 2 |
| $\boxed{\div}$ 5 $\boxed{1/x}$ | 0.2 |
| $\boxed{x^2}$ | 0.04 |
| $\boxed{=}$ | 50. |

## POWERS AND ROOTS -- $\boxed{Y^x}$

EXAMPLES: $2^6$ AND 64

| PRESS | DISPLAY |
|---|---|
| $\boxed{\text{CLR}}$ | 0 |
| 2 $\boxed{Y^x}$ 6 $\boxed{=}$ | 64 |
| $\boxed{\text{CLR}}$ | 0 |
| 64 $\boxed{\text{INV}}$ $\boxed{Y^x}$ 6 $\boxed{=}$ | 2. |

# OTHER ALGEBRAIC FUNCTIONS

LOGARITHMS -- $\boxed{\text{LMX}}$, *$\boxed{\text{LOG}}$

ANGULAR MODE KEYS -- $\boxed{\text{DEG}}$, *$\boxed{\text{RAD}}$, *$\boxed{\text{GRAD}}$

TRIGONOMETIC KEYS -- *$\boxed{\text{SIN}}$, *$\boxed{\text{COS}}$, *$\boxed{\text{TAN}}$

## CONVERSIONS

DEGREE FORMAT CONVERSIONS -- *$\boxed{\text{D.MS}}$

POLAR/RECTANGULAR CONVERSIONS -- *$\boxed{\text{P} \rightarrow \text{R}}$

# LESSON 3. CALCULATOR OPERATION USING EXISTING PROGRAMS

Objectives

- -- To become familiar with memory basics
- -- To be able to keystroke program steps into memory
- -- To be able to enter data into data registers
- -- To practice problem solution using existing programs
- -- To practice recording and reading magnetic cards and using the printer

## 3-1 BASIC PROGRAM CONTROL FUNCTIONS

$\boxed{\text{LRN}}$ - LEARN -- Pressing the learn key puts the calculator in learn mode of operation. This allows an operator to keystroke program steps into memory. Pressing the learn key again puts the calculator back under keyboard control or run mode of operation.

*$\boxed{\text{CP}}$ - CLEAR PROGRAM -- From the keyboard, the key clears the program steps, T-register, and the subroutine return register and resets all flags. When the key is encountered in the program, it zeros the T-register.

$\boxed{\text{R/S}}$ - RUN/STOP -- Reverses the status (Start/stop) of processing (run mode) when the key is encountered from the keyboard or within the program.

RST - RESET -- Resets the program pointer to location 000 of program memory, clears the subroutine return register and resets all flags.

*

Pause - PAUSE -- When encountered within the program during program execution, the current value in the display register is shown for 1/2 second. Multiple pauses causes a longer display. When encountered from the keyboard during program execution, the display shows the result from the program step.

3-2 MEMORY STORAGE CAPACITY

1. Memory is composed of two areas:

    -- Program memory locations

    -- Data memory registers

2. The memory has 120 registers for storage in the memory storage area

3. Initially, there is an even split between program and data registers with 60 for each.

4. Program memory contains eight program steps (Pressing 8 keys) for each program memory location or 480 program steps. Program steps are numbered from 000 through 479.

5. A data memory register will accept up to eight digits. If the number is greater than eight digits, the data register stores the number in scientific notation. Data registers are numbered from 00 through 59.

6. Memory storage area is divided into four banks as follows:

| Bank | Program Memory Locations | Data Memory Registers |
|---|---|---|
| 1 | 000-239 | 90-99 |
| 2 | 240-479 | 60-89 |
| 3 | 480-719 | 30-59 |
| 4 | 720-959 | 00-29 |

An understanding of banks is needed for recording on magnetic cards.

## 3-3 PARTITIONING MEMORY

1. To determine present memory partition, press keys $^{*}\boxed{\text{Op}}$ 16. When the on switch has been engaged, the memory should automatically partition at program step line 479, data register 59. When the keys $^{*}\boxed{\text{Op}}$ 16 are pressed, the display should read 479.59. The number to the left of the decimal indicates the maximum number of program steps (480). The number to the right of the decimal point indicates the maximum number of data registers (60).

2. When the number of program steps is greater than 480 or the number of data registers is greater than 60, the memory must be repartitioned. Repartitioning is set by the number of data registers. This can best be

explained with examples. Key in X $\overset{*}{\boxed{\text{Op}}}$ 17 where X represents the desired set of data registers. Each data register set has 10 data registers. Therefore if we needed 50 data registers, key in 5 $\overset{*}{\boxed{\text{Op}}}$ 17 and the memory storage area changes to 559.49. If we needed 80 data registers, key in 8 $\overset{*}{\boxed{\text{Op}}}$ 17 and the memory storage area changes to 319.79.

3. Refer back to visual 3-2 and practice other partitioning combinations.

## 3-4 LEARN MODE

1. Enter learn mode

2. Display format

$$\underbrace{000}_{\substack{\text{Program}\\\text{location}}} \qquad \underbrace{00}_{\substack{\text{Instruction}\\\text{Code}}}$$

3. Program location starts at 000 and is numbered consecutively up to the partition.

4. The instruction code contains zeros until program steps are entered.

5. Each time a key is pressed in learn mode an instruction code is entered at the program location.

## 3-5 INSTRUCTION CODES

Numeric Keys

1. Represented by the appropriate number

2. Example -- 8 is recorded as 08

First Function Keys

1. Assigned key codes based on location of the keys on the keyboard.

2. The first digit is the row location numbered from 1-9 from top to bottom.

3. The second digit is the column location numbered from 1-5 from left to right.

Second Function Keys

1. The row number is not changed.

2. The column digit is the column location plus five numbered from 6-0 from left to right.

3. Example -- *$\boxed{\text{Sin}}$ is recorded as 38.

## 3-6 ENTER PROGRAM IN MEMORY

1. Turn calculator off and on

2. Enter program

| Press | Display |
|---|---|
| *LRN | 000 00 |
| *Lbl | 001 00 |
| A | 002 00 |
| 6 | 003 00 |
| + | 004 00 |
| 9 | 005 00 |
| = | 006 00 |
| R/S | 007 00 |
| LRN | 0 |

3. Run program

| Press | Display |
|---|---|
| A | 15 |

## 3-7 EDITING A PROGRAM

- Use previous example

- Keys -- The first four keys do not register an instruction code when pressed in learn mode.

  SST -- Moves line counter to next program step

  BST -- Moves line counter back one step.

  *Ins -- Inserts a new line at display point

  *Del -- Deletes the displayed line

  *Nop -- Deletes instruction and provides a blank space (No operation code)

- Correcting an error on any line can be completed by moving the program pointer to the incorrect line and pressing the correct key (Will not work for keystroke code exceptions)

## 3-8 COMBINED INSTRUCTION CODES

1. When two or more keys are used for a single instruction, the key code is not based on key location.

2. A single location code number is assigned for combined instructions. A numeric key location is used for this purpose.

3. Combined instruction codes

| Key Sequence | | Key Codes |
|---|---|---|
| *Pgm | *Ind | 62 |
| *Exc | *Ind | 63 |
| *Prd | *Ind | 64 |
| STO | *Ind | 72 |
| RCL | *Ind | 73 |
| SUM | *Ind | 74 |
| GTO | *Ind | 83 |
| *Op | *Ind | 84 |
| INV | SBR | 92 |

4. 2nd key -- Combined with other instructions.

## 3-9 TWO-DIGIT ADDRESSES

1. Two-digit addresses following select keys occupy one location.

2. Instructions for:

   -- Data memory (STO, RCL, SUM, Prd, Exc)

   -- Program library access (Pgm)

   -- Special control operations (Op)

3. Examples -- $\boxed{\text{STO}}$ 25 is recorded as:
   ⌊42⌋, ⌊25⌋rather than ⌊42⌋, ⌊02⌋, ⌊05⌋

## 3-10 TRANSFER ADDRESSES

1. Transfer addresses to a three digit program location are stored in two locations with the first digit stored in the first location and the remaining two digits stored in the second location.

2. Transfer addresses (GTO, SBR, x=t, x≥t, Dsz, If Flg)

3. Example -- $\boxed{\text{GTO}}$ 125 is recorded as:
   ⌊61⌋, ⌊01⌋, ⌊25⌋rather than ⌊61⌋, ⌊01⌋, ⌊02⌋, ⌊05⌋

## 3-11 DATA ENTRY PROCEDURES

- Function -- Store a number in a data register

- Methods

  1. Direct -- 3 $\boxed{\text{STO}}$ 01 stores a 3 in data register 01 and 5 $\boxed{\text{STO}}$ 02

  2. Programmed -- Create steps in the program to store a displayed number

- Example of programmed method

  1. Input program steps

     ```
     Lbl A
     STO 01
     R/S
     Lbl B
     STO 02
     R/S
     ```

  2. Input numbers

     ```
     3 A
     5 B
     ```

- Use recall key to check if numbers are in the assigned data registers.

## 3-12 LOAN PAYMENT PROGRAM LIST

1. Problem -- Calculate the annual payment on an equal payment amortized loan given the interest rate, principal borrowed and the number of annual payments

2. Method

$$A = \frac{r}{1 - \left(\frac{1}{1+r}\right)^n}$$

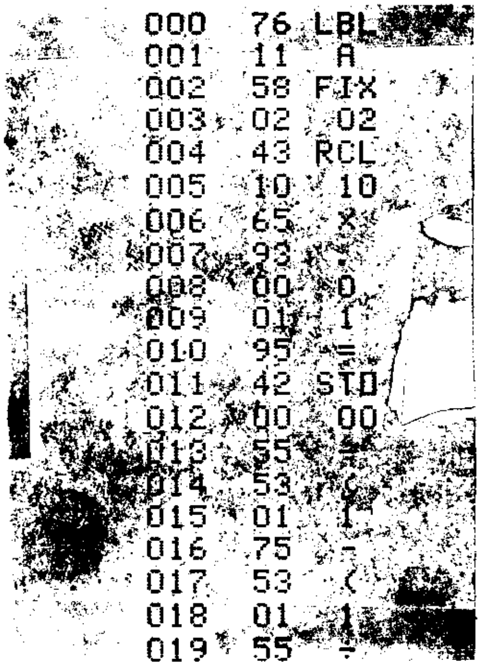
Where:

$A$ = annual payment for a $1 loan

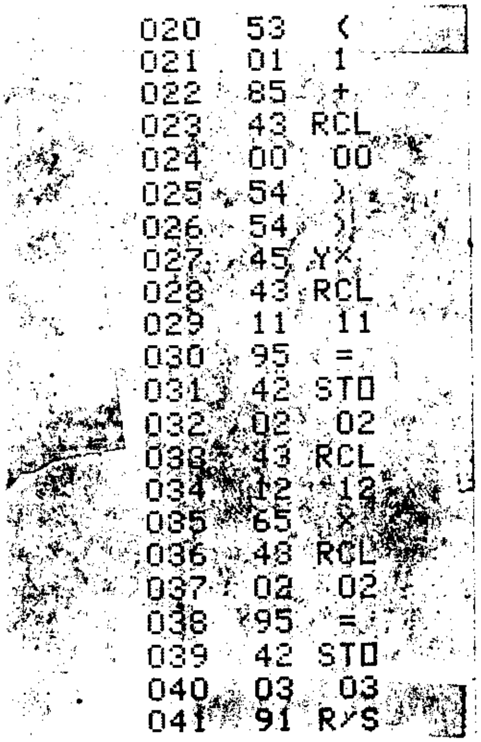
$r$ = annual interest rate

$n$ = number of payments

3. Program List



```
000  76 LBL
001  11  A
002  58 FIX
003  02  02
004  43 RCL
005  10  10
006  65  X
007  93  .
008  00  0
009  01  1
010  95  =
011  42 STO
012  00  00
013  55  ÷
014  53  (
015  01  1
016  75  -
017  53  (
018  01  1
019  55  ÷
```



```
020  53  (
021  01  1
022  85  +
023  43 RCL
024  00  00
025  54  )
026  54  )
027  45 YX
028  43 RCL
029  11  11
030  95  =
031  42 STO
032  02  02
033  43 RCL
034  12  12
035  65  X
036  43 RCL
037  02  02
038  95  =
039  42 STO
040  03  03
041  91 R/S
```

## 3-13 LOAN PAYMENT PROGRAM -- RUN MODE

Input Data

| | | |
|---|---|---|
| 1. | Interest rate in percent | 15 STO 10 |
| 2. | Number of annual payments | 3 STO 11 |
| 3. | Amount of loan | 1000 STO 12 |

Output

1. Press A
2. Answer $437.98 payment per year

## 3-14 MEMORY AREA AND MAGNETIC CARD RECORDING

The 120 memory registers of the calculator are split into 4 banks of 30. Each card records 2 banks, one to a side.

Therefore, each side is capable of storing 240 program steps or 30 data registers. The program steps and data registers are recorded on the cards in a specific manner, although the order in which they are recorded does not matter.

Assume a program contains 960 steps (the maximum) and no data registers:

| | |
|---|---|
| Card side one records steps | 000 to 239 |
| Card side two records steps | 240 to 479 |
| Card side three records steps | 480 to 719 |
| Card side four records steps | 720 to 959 |

Assume a program contains 160 steps and 100 data registers (the maximum):

| | |
|---|---|
| Card side four records data registers | 00 to 29 |
| Card side three records data registers | 30 to 59 |
| Card side two records data registers | 60 to 89 |
| Card side one records data registers | 90 to 99 |
| and steps | 000 to 159 |

## 3-15 RECORDING AND READING MAGNETIC CARDS

Recording

1. Use previous example
2. Remove Fix, EE or Eng
3. Press 1 *Write
4. Insert card side 1 in the slot at the right side of the calculator
5. Remove card from left side
6. Identify card
7. Flashing number -- An error

Reading

1. Turn calculator off and on
2. Insert card side 1 in the slot at the right side of the calculator
3. Remove card
4. Bank number shows in the display
5. Operate the program -- Visual 3-13
6. Flashing number -- Card not properly read

## 3-16 PRINTER CONTROL

Printer Keys

Print -- Display value is printed when pressed

Trace -- Down position results in every step of a calculation to be printed

ADV ↑ -- Advances paper

Keyboard

RST *List -- Program list

RST INV *LIST -- List contents of data registers

R/S -- Stop listing

## BASIC PROGRAM CONTROL FUNCTIONS

LRN -- LEARN

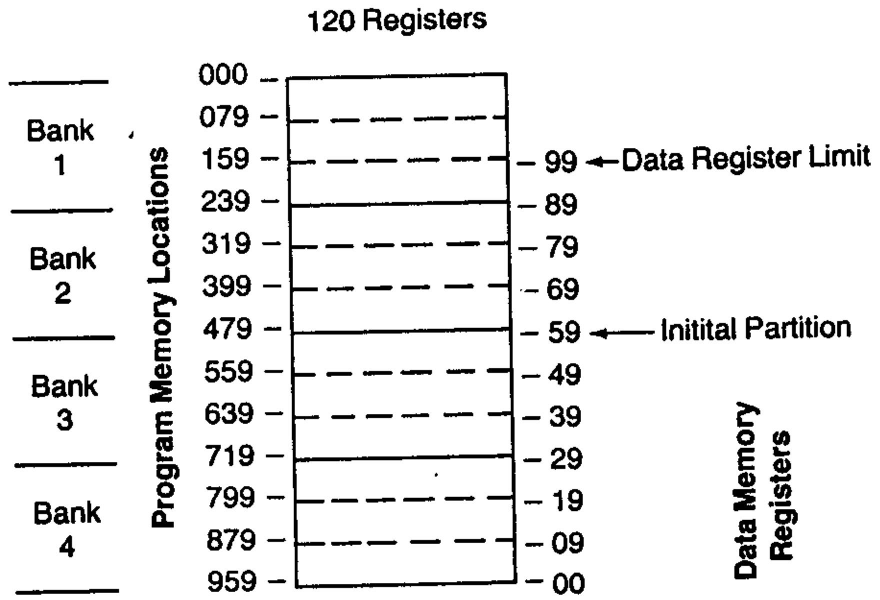
*CP -- CLEAR PROGRAM

R/S -- RUN/STOP

RST -- RESET

*PAUSE -- PAUSE

# MEMORY STORAGE CAPACITY



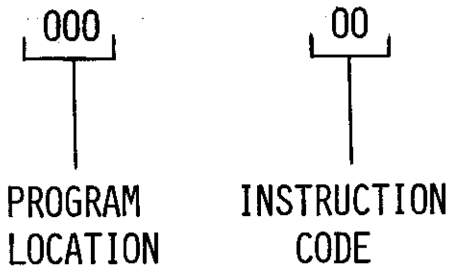
TI Programmable 59

## PARTITIONING MEMORY

| ACTION | PRESS | DISPLAY |
|---|---|---|
| CHECK CURRENT PARTITION | *[Op] 16 | 479.59 |
| PARTITION STORAGE AREA FOR 50 DATA REGISTERS | 5 *[Op] 17 | 559.49 |
| PARTITION STORAGE AREA FOR 80 DATA REGISTERS | 8 *[Op] 17 | 319.79 |

## LEARN MODE

## DISPLAY FORMAT



## Keyboard And Instruction Key Codes

| Key | Key Code | Key | Key Code | Key | Key Code | Key | Key Code | Key | Key Code |
|---|---|---|---|---|---|---|---|---|---|
| A' | 16 | B' | 17 | C' | 18 | D' | 19 | E' | 10 |
| A | 11 | B | 12 | C | 13 | D | 14 | E | 15 |
|  |  | INV | 27 | log | 28 | CP | 29 | CLR | 20 |
| 2nd | Merged | INV | 22 | lnx | 23 | CE | 24 | CLR | 25 |
| Pgm | 36* | P→R | 37 | sin | 38 | cos | 39 | tan | 30 |
| LRN | None | x⇄t | 32 | $x^2$ | 33 | $\sqrt{x}$ | 34 | 1/x | 35 |
| Ins | None | CMs | 47 | Exc | 48* | Prd | 49* | Ind | 40 (or merged) |
| SST | None | STO | 42* | RCL | 43* | SUM | 44* | $y^x$ | 45 |
| Del | None | Eng | 57 | Fix | 58* | Int | 59 | \|x\| | 50 |
| BST | None | EE | 52 | ( | 53 | ) | 54 | ÷ | 55 |
| Pause | 66 | x=t | 67* | Nop | 68 | Op | 69* | Deg | 60 |
| GTO | 61* | 7 | 07 | 8 | 08 | 9 | 09 | × | 65 |
| Lbl | 76* | x≥t | 77* | Σ+ | 78 | $\bar{x}$ | 79 | Rad | 70 |
| SBR | 71* | 4 | 04 | 5 | 05 | 6 | 06 | − | 75 |
| St flg | 86* | If flg | 87* | D.MS | 88 | π | 89 | Grad | 80 |
| RST | 81 | 1 | 01 | 2 | 02 | 3 | 03 | + | 85 |
| Write | 96 | Dsz | 97* | Adv | 98 | Prt | 99 | List | 90 |
| R/S | 91 | 0 | 00 | • | 93 | +/− | 94 | = | 95 |

ENTER PROGRAM IN MEMORY

| PRESS | DISPLAY |
|---|---|
| LRN | 000 00 |
| *LBL | 001 00 |
| A | 002 00 |
| 6 | 003 00 |
| + | 004 00 |
| 9 | 005 00 |
| = | 006 00 |
| R/S | 007 00 |
| LRN | 0 |
| A | 15 |

## EDITING A PROGRAM

| ACTION | PRESS | DISPLAY |
|---|---|---|
| | CLR | 0 |
| | RST | 0 |
| | LRN | 000 76 |
| | SST | 001 11 |
| | BST | 000 76 |
| MOVE TO LINE 003 | SST | 003 85 |
| | *INS | 003 00 |
| | *DEL | 003 85 |

# COMBINED INSTRUCTION CODES

| Key | Key Code | Key | Key Code | Key | Key Code | Key | Key Code | Key | Key Code |
|---|---|---|---|---|---|---|---|---|---|
| A' | 16 | B' | 17 | C' | 18 | D' | 19 | E' | 10 |
| A | 11 | B | 12 | C | 13 | D | 14 | E | 15 |
|  |  | INV | 27 | log | 28 | CP | 29 | CLR | 20 |
| 2nd | Merged | INV | 22 | lnx | 23 | CE | 24 | CLR | 25 |
| Pgm | 36* | P→R | 37 | sin | 38 | cos | 39 | tan | 30 |
| LRN | None | x:t | 32 | $x^2$ | 33 | $\sqrt{x}$ | 34 | $1/x$ | 35 |
| Ins | None | CMs | 47 | Exc | 48* | Prd | 49* | Ind | 40 (or merged) |
| SST | None | STO | 42* | RCL | 43* | SUM | 44* | $y^x$ | 45 |
| Del | None | Eng | 57 | Fix | 58* | Int | 59 | $\|x\|$ | 50 |
| BST | None | EE | 52 | ( | 53 | ) | 54 | ÷ | 55 |
| Pause | 66 | x=t | 67* | Nop | 68 | Op | 69* | Deg | 60 |
| GTO | 61* | 7 | 07 | 8 | 08 | 9 | 09 | × | 65 |
| Lbl | 76* | $x \geq t$ | 77* | Σ+ | 78 | $\bar{x}$ | 79 | Rad | 70 |
| SBR | 71* | 4 | 04 | 5 | 05 | 6 | 06 | − | 75 |
| St flg | 86* | If flg | 87* | D.MS | 88 | π | 89 | Grad | 80 |
| RST | 81 | 1 | 01 | 2 | 02 | 3 | 03 | + | 85 |
| Write | 96 | Dsz | 97* | Adv | 98 | Prt | 99 | List | 90 |
| R/S | 91 | 0 | 00 | • | 93 | +/− | 94 | = | 95 |

## TWO-DIGIT ADDRESSES

DATA MEMORY

STO XX

RCL XX

SUM XX

*PRD XX

*EXC XX

PROGRAM LIBRARY ACCESS -- -- PGM XX

SPECIAL CONTROL OPERATIONS -- OP XX

TRANSFER ADDRESSES

$\boxed{\text{GTO}}$ XXX

$\boxed{\text{SBR}}$ XXX

*$\boxed{x = T}$ XXX

*$\boxed{x \geq T}$ XXX

*$\boxed{\text{Dsz}}$ XXX

*$\boxed{\text{If flg}}$ XXX

## DATA ENTRY PROCEDURES

1. DIRECT 3 $\boxed{\text{STO}}$ 01

   5 $\boxed{\text{STO}}$ 02

2. PROGRAMMED

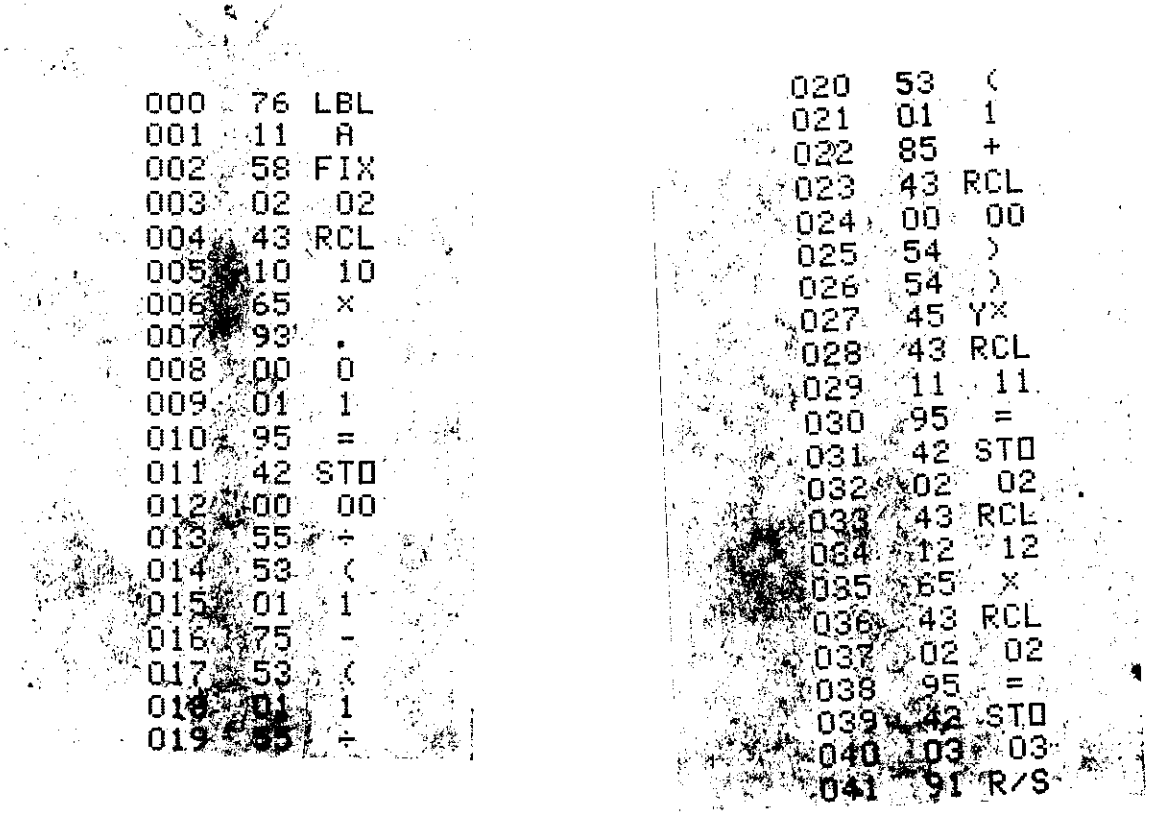
   INPUT PROGRAM

   ```
   LBL A
   STO 01
    R/S
   LBL B
   STO 02
    R/S
   ```

3. INPUT NUMBERS

   ```
   3
   A
   5
   B
   ```

# LOAN PAYMENT PROGRAM LIST



```
000  76  LBL
001  11   A
002  58  FIX
003  02   02
004  43  RCL
005  10   10
006  65   ×
007  93   .
008  00   0
009  01   1
010  95   =
011  42  STO
012  00   00
013  55   ÷
014  53   (
015  01   1
016  75   -
017  53   (
018  01   1
019  55   ÷
```

```
020  53   (
021  01   1
022  85   +
023  43  RCL
024  00   00
025  54   )
026  54   )
027  45  YX
028  43  RCL
029  11   11
030  95   =
031  42  STO
032  02   02
033  43  RCL
034  12   12
035  65   X
036  43  RCL
037  02   02
038  95   =
039  42  STO
040  03   03
041  91  R/S
```

## LOAN PAYMENT PROGRAM -- RUN MODE

INPUT DATA

1. INTEREST RATE IN PERCENT: 15 $\boxed{\text{STO}}$ 10

2. NUMBER OF ANNUAL PAYMENTS: 3 $\boxed{\text{STO}}$ 11

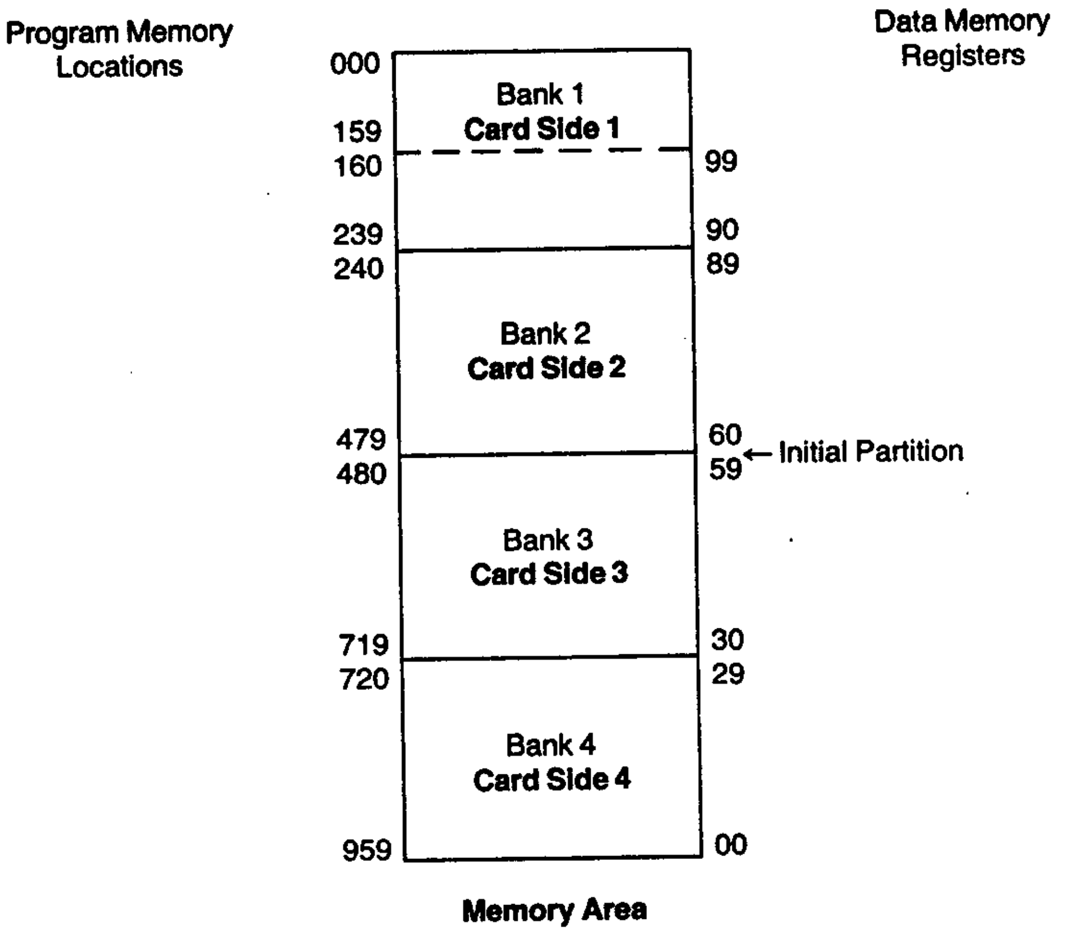
3. AMOUNT OF LOAN: 1000 $\boxed{\text{STO}}$ 12

OUTPUT

1. PRESS $\boxed{\text{A}}$

2. DISPLAY: 437.98

## MEMORY AREA AND MAGNETIC CARD RECORDING



# RECORDING AND READING MAGNETIC CARDS

## RECORDING

1. PRESS 1 $^{*}\boxed{\text{WRITE}}$
2. INSERT CARD

## READING

1. TURN CALCULATOR OFF AND ON
2. INSERT CARD
3. OPERATE THE LOAN PAYMENT PROGRAM

# PRINTER CONTROL

## PRINTER KEYS

PRINT -- PRINT DISPLAY VALUE

TRACE -- TRACE CALCULATION

ADV 🡅 -- ADVANCE PAPER

## KEYBOARD

RST *LIST -- PROGRAM LIST

RST INV *LIST -- DATA REGISTER LIST

R/S -- STOP LISTING

## LESSON 4. PROGRAMMING BASICS

Objectives

-- To understand the mechanics of programming

-- To learn how to structure a problem and develop a sequence for problem solution

-- To be able to use labels, variables, transfers and addresses in a program

### 4-1 MECHANICS OF PROGRAMMING

The versatile arithmetic language permits both simple and complex programming. Simple programs may be entered, checked, and run with little effort or difficulty. Even though the language is designed to be as straightforward as possible a complex program requires forethought and planning.

If you have done little programming, you will find the following ideas useful. If you are familiar with programming concepts, the ideas will serve as a review and orient you toward calculator programming. You should interpret the following only as a list of suggestions since you will undoubtedly develop your own programming style.

1. **Define the problem very clearly and carefully.** Identify the formulas, variables and desired results. What is known? What is to be determined? How are the known and the unknown related?
2. **Develop a method of solution** (sometimes called an **algorithm**). Define the operation sequence of the numerical approach you want to use keeping in mind the calculating and programming capabilities of the calculator. (Remember, strictly speaking, calculators do not solve problems, you do. Your calculator carries out your solutions precisely the way you tell it to!)
3. **Develop a flow diagram.** It is often useful to develop drawings that help you visualize the flow of the program. Here, you can picture interactions between various parts of the solution. It may even be possible to simplify the program structure after it is flow charted.
4. **Begin making data register assignments.** Assign data registers to the numerous things you'll be operating on. You'll continue this task throughout the programming process. It is a good idea to never store a quantity in memory without making a written note that the data register in question contains that quantity.
5. **Translate the flow diagram into keystrokes.** The coding forms are provided to help you here. It is useful to list all labels and memory registers in the space indicated on the form. Use the comments column for easy reference to various segments of the program.
6. **Enter the program.** Press [2nd] [CP] [LRN] and key in the complete program from the coding form. When entry is complete, press [LRN] to remove the calculator from the learn mode.
7. **Test the program.** Check out the program using test problems representing as many cases as practical.
8. **Correct errors.** Correct the coding form for any errors discovered while testing the program.

9. **Edit the program.** Place the calculator in the learn mode, complete the required corrections and press $\boxed{\text{LRN}}$ to return to the keyboard operation. See page IV-21 for more information.
10. **Retest the program.** Repeat steps 7-9 as needed.
11. **Record the program.** If your calculator has magnetic card capability, record the program on magnetic cards. See Section VII for more information.
12. **Document user instructions.** It's always a good idea to carefully write down step-by-step instructions describing how to use your program. Even the most powerful programs are useless if you don't remember how to use them. Fill out a User Instructions form, detailing information required to run the program.

## 4-2 PROGRAM EXECUTION FLOW -- NORMAL

- Program execution starts with program location 000 and sequentially performs the calculations until a R/S is encountered.
- The sequence for problem solution and, therefore, the program steps must be sequentially from beginning to end.
- The normal flow of a program is changed with transfers and addresses.
- Intermediate calculations used as input for latter calculations must be performed first.

## 4-3 LABELS

1. Labels provide an easy access to any location within a program. Labels are reference points when used within a program.
2. User-defined labels -- $\boxed{\text{A}}$ - $\boxed{\text{E}}$ , *$\boxed{\text{A}}$ - *$\boxed{\text{E}}$

3. When a user-defined label is pressed from the keyboard, the program pointer searches the program list starting with 000 until the label is located. Program execution starts with the first instruction following the label.

4. Common labels -- all remaining keys except: 2nd, SST, lns, LRN, BST, Del, Ind, and numbers 0-9.

5. Common labels can only be used within a program as a reference point. Pressing a common label key from the keyboard only performs that operation.

6. When common labels are used within a program as a reference point, the operation (or definition of the key) is not acted upon in the program.

7. A label key within a program must be preceded by the key Lbl.

8. A label may only be used once within a program segment.

## 4-4 VARIABLES IN A PROGRAM

- Rather than specify data in a program (3 and 5), a variable may be included in the program and recalled from a data register.

- Form -- RCL XX

- Calculations made within a program can be stored in a data register.

- Form -- STO XX

## 4-5 TRANSFERS

- Unconditional transfers branch the program flow to the label or the program location.

- Conditional transfers test a value and transfer to a label or program location based on the test value.

## 4-6 UNCONDITIONAL TRANSFERS

- GTO n or xxx -- Diverts the flow of processing to label n or program location xxx when encountered in a program -- From the keyboard, the program location printer is positioned at label n or program location xxx.

- RST -- Resets pointer at program location 000 when encountered in a program or from the keyboard.

- SBR n or xxx -- From the keyboard, the program location pointer is positioned at label n or program location xxx -- When encountered in a program, it diverts the flow of processing to label n or program location xxx, executes instructions at that point until a RETURN (INV SBR) instruction is located and returns to the original point in the main program.

## 4-7 CONDITIONAL TRANSFERS

1. Instructions that are capable of making decisions within a program -- A transfer is made depending upon the outcome of a test.

2. Four types of test questions:
   a. Is one number equal to or greater than the other?
   b. Is one number less than another?
   c. Are the two numbers equal?
   d. Are the two numbers unequal?

3. Types of conditional transfer instructions:
   a. Compare display register contents to T - register: $x = t$, $x \geq t$
   b. Test contents of data register 0-9: Dsz
   c. Test status of program flags: lf flg

4. A transfer address follows each of these instructions. When the answer to the test is "yes" (test positive), transfer is made to that address. If the answer to the test is "no" (test negative), the transfer is skipped.

## 4-8 T - REGISTER

- Special data memory register
- Used to "Test" a number
- Number is placed in the T - register with the "exchange" key: $\boxed{x \leftrightarrows t}$
- Activated from keyboard or within a program

## 4-9 EQUAL TO OR GREATER THAN TEST

1. $\overset{*}{\boxed{x \geq t}}$ n or xxx -- This command compares the display value with the "T" value. If the display value "x" is greater than or equal to the "T" value, the program segment transfers to the label n or program location xxx. If the display value "x" is not greater than or equal to "T," the program skips the location transfer and continues with the next instruction.

2. Key sequence

   T value

   Exchange key

   x value

   Test -- $\overset{*}{\boxed{x \geq t}}$

   Label or program location

   Next instruction

3. $\boxed{\text{INV}}$ $\overset{*}{\boxed{x \geq t}}$ -- Reverses the answer

## 4-10 EQUAL TO TEST

1. $\overset{*}{\boxed{x = t}}$ n or xxx -- This command compares the display value with the "T" value. If the display value "x" is equal to the "T" value, the program segment transfers to the label n or program location xxx. If the display value "x" is not equal to "T", the program skips the location transfer and continues with the next instruction.

2. Key sequence

   T value

   Exchange key

   x value

   Test -- $\overset{*}{\boxed{x = t}}$

   Label or program location

   Next instruction

3. $\boxed{\text{INV}}$ $\overset{*}{\boxed{x = t}}$ -- Reverses the answer

## 4-11 CONDITIONAL TRANSFER EXAMPLE

1. A borrower can obtain a loan from a financial institution with simple interest. The interest rate on the loan varies with the length of the repayment period as follows:

   a. Repayment period less than one year

   b. Repayment period one year or more, but less than ten years, or

   c. Repayment period ten years or more

2. Problem -- Determine the amount of interest paid on the first year of the loan

3. Programming process

   a. Assign the data registers with the variable input

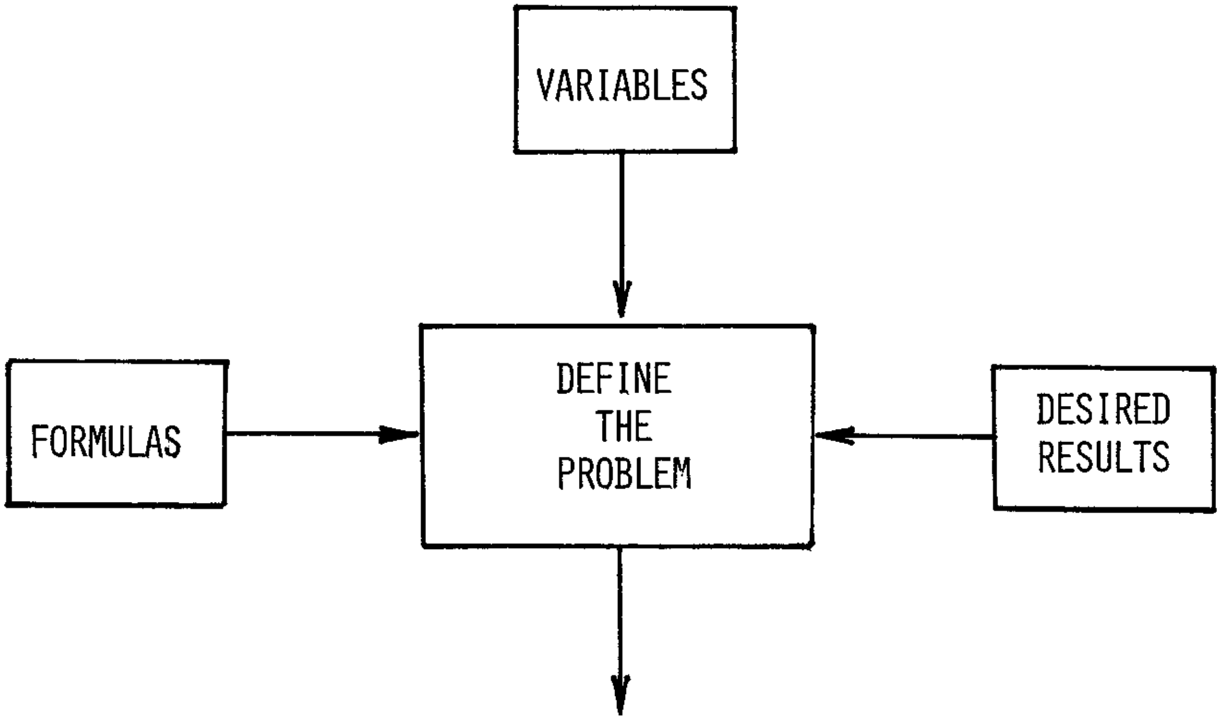
   b. Write the program instructions

   c. Key program instructions into memory

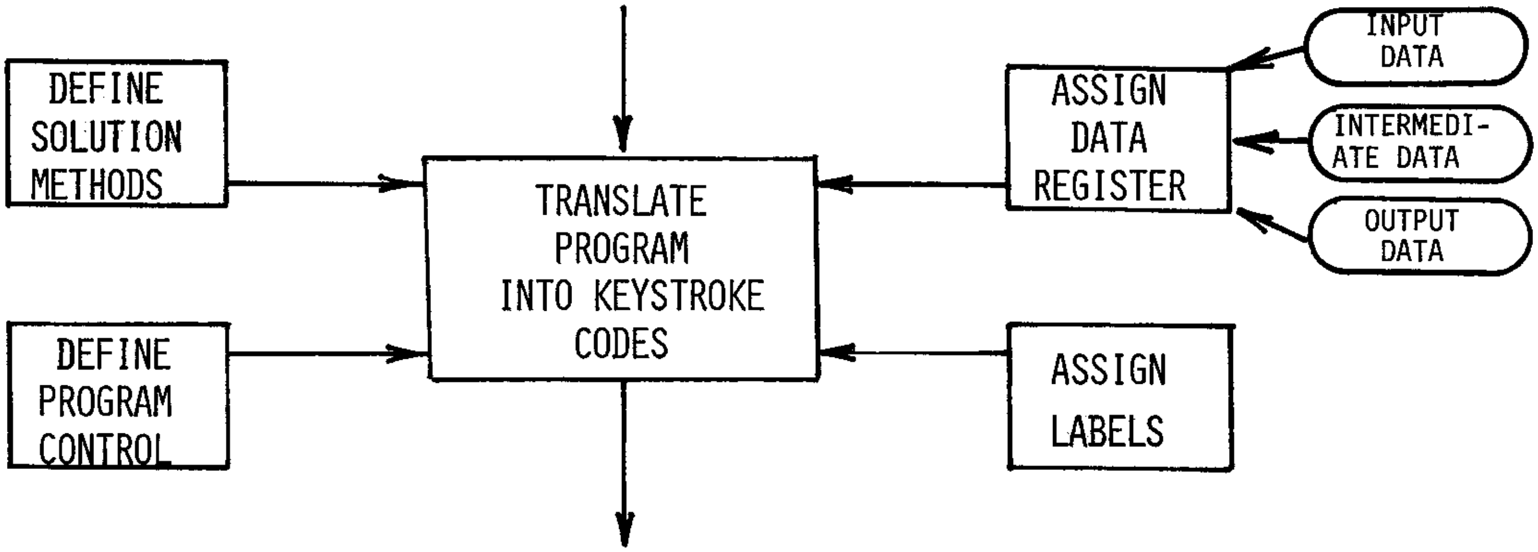
   d. Enter variable data

   e. Run program

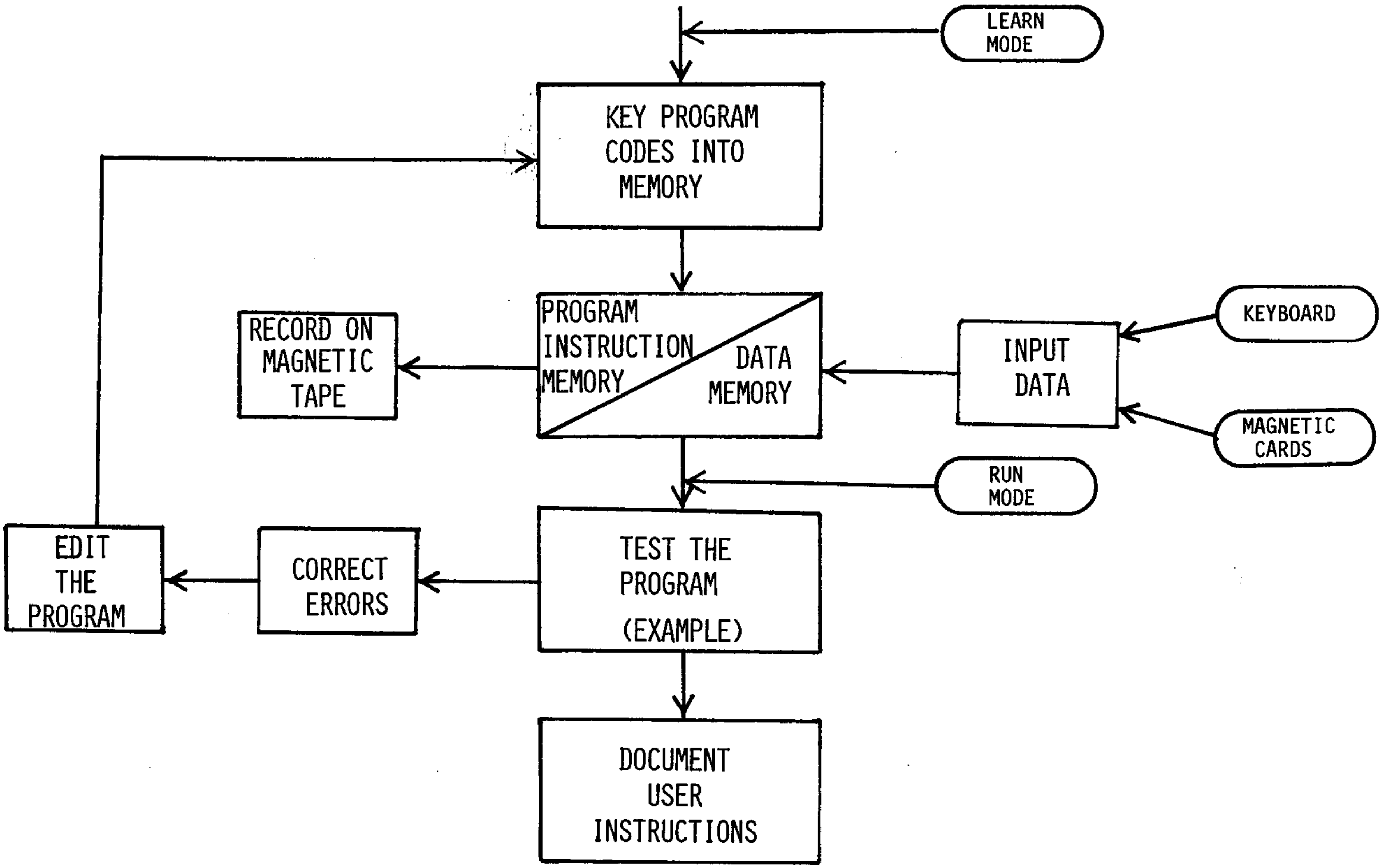
# MECHANICS OF PROGRAMMING

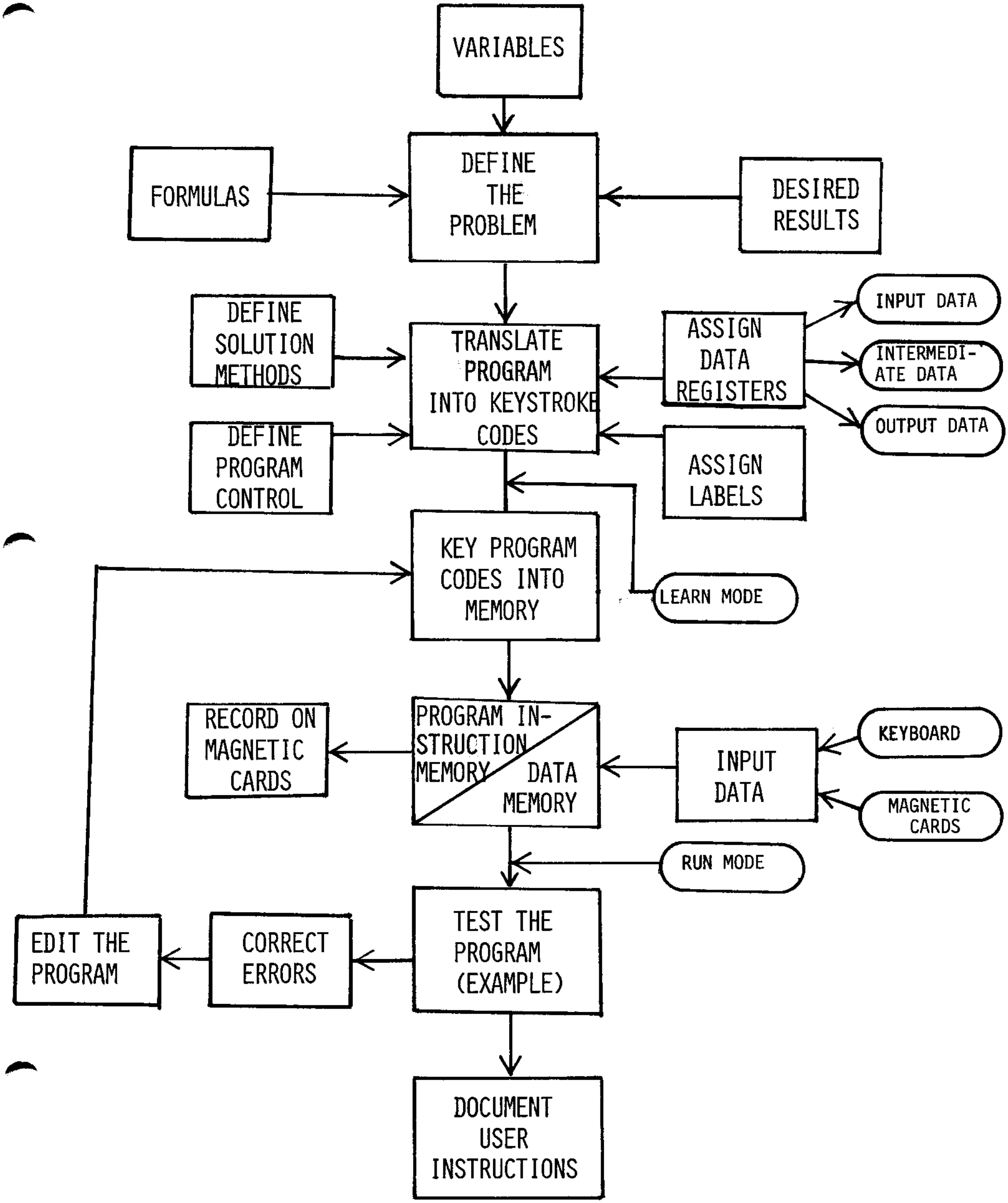


## MECHANICS OF PROGRAMMING (CONT.)

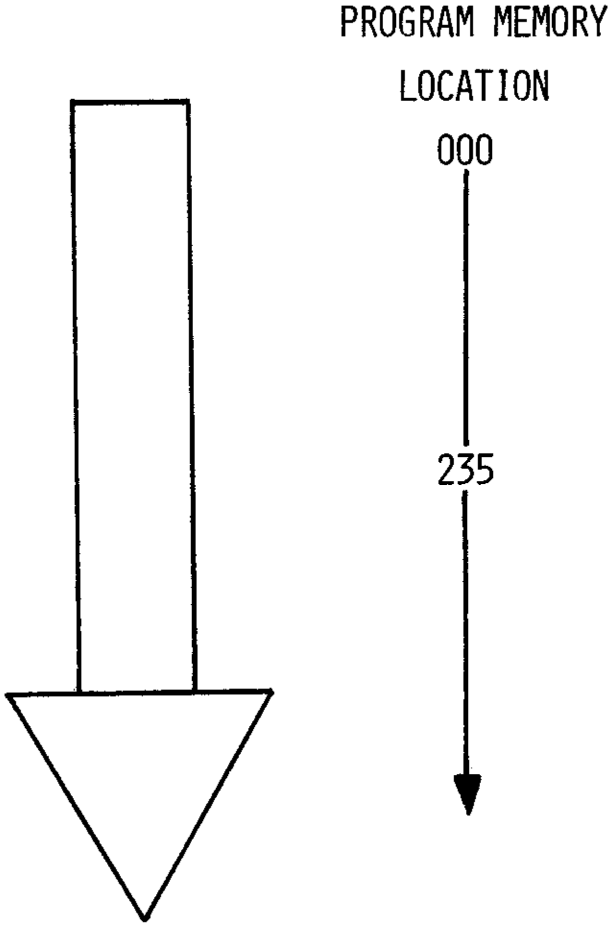


## MECHANICS OF PROGRAMMING (CONT.)




VARIABLES
FORMULAS
DEFINE THE PROBLEM
DESIRED RESULTS
DEFINE SOLUTION METHODS
TRANSLATE PROGRAM INTO KEYSTROKE CODES
ASSIGN DATA REGISTERS
INPUT DATA
INTERMEDI- ATE DATA
OUTPUT DATA
DEFINE PROGRAM CONTROL
ASSIGN LABELS
KEY PROGRAM CODES INTO MEMORY
LEARN MODE
RECORD ON MAGNETIC CARDS
PROGRAM IN- STRUCTION MEMORY
DATA MEMORY
INPUT DATA
KEYBOARD
MAGNETIC CARDS
RUN MODE
EDIT THE PROGRAM
CORRECT ERRORS
TEST THE PROGRAM (EXAMPLE)
DOCUMENT USER INSTRUCTIONS

## PROGRAM EXECUTION FLOW -- NORMAL



## LABELS

- USER-DEFINED LABELS -- $\boxed{\text{A}}$ — $\boxed{\text{E}}$, $^*\boxed{\text{A}}$ — $^*\boxed{\text{E}}$

- COMMON LABELS -- ALL KEYS EXCEPT:

  $\boxed{\text{2ND}}$, $\boxed{\text{SST}}$, $^*\boxed{\text{INS}}$, $\boxed{\text{LRN}}$, $\boxed{\text{BST}}$, $^*\boxed{\text{DEL}}$, $^*\boxed{\text{IND}}$

  AND THE NUMERICAL KEYS $\boxed{0}$ THROUGH $\boxed{9}$

- EXAMPLE PROGRAM

  LBL A

  3 + 5 =

  R/S

- RUN MODE -- PRESS A

  ANSWER 8

## VARIABLES IN A PROGRAM

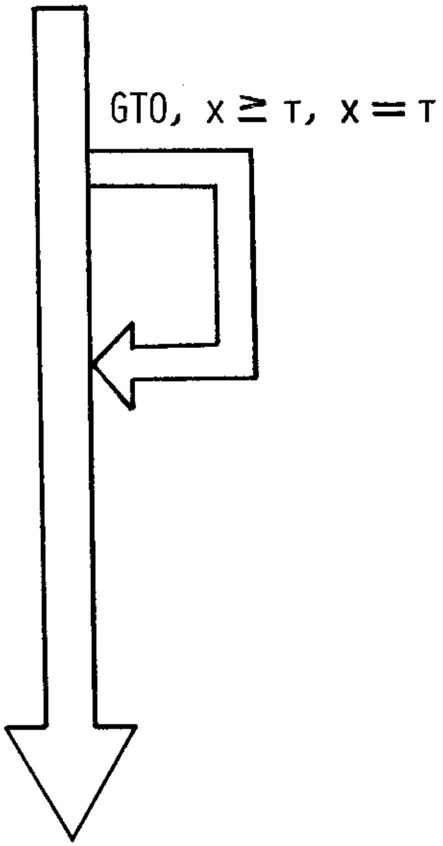
- EXAMPLE PROGRAM

```
LBL A
RCL 01 + RCL 02 = STO 10
R/S
```

- RUN MODE
  1. ENTER NUMBERS IN DATA
     REGISTER 01 AND 02
  2. PRESS A
  3. WHAT IS YOUR ANSWER?

## PROGRAM EXECUTION FLOW

## TRANSFERS



## UNCONDITIONAL TRANSFERS

- GO TO INSTRUCTION -- $\boxed{\text{GTO}}$ N OR xxx
- RESET INSTRUCTION -- $\boxed{\text{RST}}$
- SUBROUTINE -- $\boxed{\text{SBR}}$ N OR xxx
- EXAMPLE PROGRAM

```
LBL SUM
+ 4 =
PAUSE
PAUSE
PAUSE
GTO SUM
```

- RUN MODE -- PRESS

```
RST
R/S
```

- WHAT IS YOUR ANSWER?

# CONDITIONAL TRANSFERS

TEST KEYS

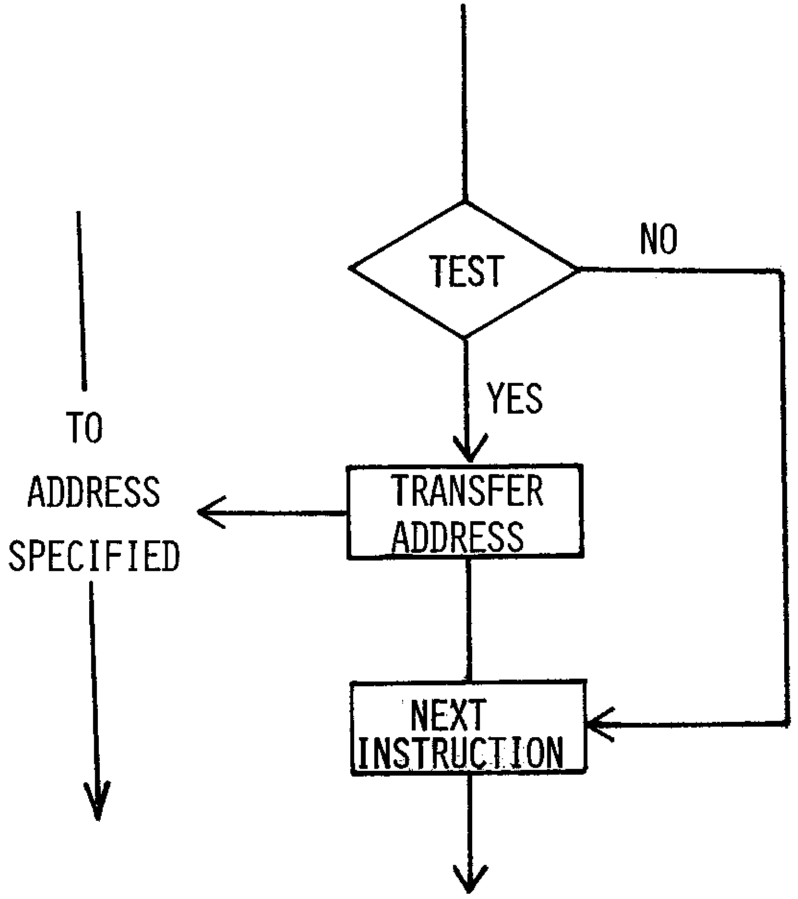
*X = T

*X ≥ T

*Dsz

*If FLG

PROGRAM EXECUTION FLOW



# T-REGISTER

- DATA MEMORY REGISTER -- T
- EXCHANGE KEY -- $\boxed{x \leftrightarrows t}$
- EXAMPLES -- CALCULATOR MODE

  5

  x⇆t

  4

  x⇆t
- WHAT NUMBER IS IN THE DISPLAY AFTER YOU PRESS

  THE EXCHANGE KEY?

## EQUAL TO OR GREATER THAN TEST

- *$\boxed{X \geq T}$ N OR XXX

- POSITIVE TEST EXAMPLE

6 IS 7 ≥ 6?

X ⇆ T

7 OR

X ≥ T YES -- TRANSFER TO LABEL $\boxed{\text{SIN}}$ AND CONTINUE EXECUTION AT THAT POINT

SIN

RCL

- NEGATIVE TEST EXAMPLE

10 IS 3 ≥ 10?

X ⇆ T

3 OR

X ≥ T NO -- CONTINUE EXECUTION WITH THE INSTRUCTION $\boxed{\text{RCL}}$

SIN

RCL

- $\boxed{\text{INV}}$ *$\boxed{X \geq T}$ N OR XXX

# EQUAL TO TEST

- $\boxed{X = T}$* N OR XXX

- POSITIVE TEST EXAMPLE

| | | |
|---|---|---|
| 14 | | IS 14 = 14? |
| X = T | | |
| 14 | <u>OR</u> | |
| X = T | | YES -- TRANSFER TO PROGRAM LOCATION 324 AND CONTINUE EXECUTION AT THAT POINT |
| 324 | | |
| RCL | | |

- NEGATIVE TEST EXAMPLE

| | | |
|---|---|---|
| 23 | | IS 26 = 23? |
| X ⇆ T | | |
| 26 | <u>OR</u> | |
| X = T | | NO -- CONTINUE EXECUTION WITH THE INSTRUCTION $\boxed{RCL}$ |
| 324 | | |
| RCL | | |

- $\boxed{INV}$ $\boxed{X = T}$* N OR XXX

## CONDITIONAL TRANSFER EXAMPLE

1. ASSIGN DATA REGISTERS

   AMOUNT OF LOAN -- REGISTER 10

   LOAN REPAYMENT PERIOD (YEARS) -- REGISTER 11

   INTEREST RATE (%) -- LOAN LESS THAN 1 YEAR -- REGISTER 12

   INTEREST RATE (%) -- LOAN 1 TO 10 YEARS -- REGISTER 13

   INTEREST RATE (%) -- LOAN 10 YEARS OR MORE -- REGISTER 14

   FIRST YEAR INTEREST -- REGISTER 20

2. PROGRAM INSTRUCTION

```
LBL A
RCL 11  x⇆t
1
INV x≥t  SIN
RCL 10 x RCL 12 x .01 x RCL 11 = STO 20
R/S
GTO COS
LBL SIN
10
```

## CONDITIONAL TRANSFER EXAMPLE (CONT)

```
    X≥T  TAN
    RCL 10 x RCL 14 x .01=STO 20
    R/S
    GTO cos
    LBL TAN
    RCL 10 x RCL 13 x .01=STO 20
    LBL cos
    R/S

3.  INPUT DATA -- RUN MODE
    10000 STO 10
       .5 STO 11
       18 STO 12
       15 STO 13
       12 STO 14

4.  PRESS A
    WHAT IS YOUR ANSWER?
5.  CHANGE REGISTER 11 TO 7
    WHAT IS YOUR ANSWER?
6.  CHANGE REGISTER 11 TO 20
    WHAT IS YOUR ANSWER?
```

# LESSON 5. ADVANCED PROGRAMMING TECHNIQUES

Objectives

-- To learn how to use loops in a program

-- To understand the use and programming of subroutines, indirect instructions, and other techniques

-- To practice programming and operating advanced programming techniques

## 5-1 LOOPING

1. Loop -- Programming technique to allow program execution to perform a sequence of instructions many times until a calculation is complete. To create a loop, the program includes an instruction that transfers the program pointer to an earlier location.

2. Unconditional Transfer Loop -- The use of the following keys:

   RST -- loops back to program location 000

   GTO -- loops back to instruction program location or label.

   An unconditional loop must include a test and transfer out of the loop or the program will not terminate.

3. Conditional Transfer Loop -- Creation of a loop where it is know how many times a calculation should be made.

A conditional loop uses the *Dsz key -- decrement and skip on zero.

## 5-2 LOOPING WITH CONDITIONAL TRANSFER

- Statement

  Dsz X, n or xxx

  Where:

  X = Data register 0-9

  n = Label

  xxx - Program location

## 5-3 LOOP EXAMPLE

- Key Sequence

  1. Initialize data registers
  2. Label for the loop address
  3. Calculations to be made
  4. Decrement key
  5. Data register counter number
  6. Label or program location

- Change the example to sum the numbers from 1 to X. What program steps need changing? What additions are needed?

  -- Store a number in register 10

  -- RCL 10 STO 00

## 5-4 LOOP USING OPERATION CODES

1. Operation codes -- Operation codes that have special functions built into the calculator.

2. General Form -- $^{*}\boxed{\text{Op}}$ nn; where the nn represents a two digit number.

3. Increment/Decrement Data registers -- Op 20-29 and Op 30-39.

4. Increment -- To increment register n by 1, press $^{*}\boxed{\text{Op}}$ 2n, where n is a data register number 0-9.

5. Decrement -- To decrement register n by 1, press $^{*}\boxed{\text{Op}}$ 3n, where n is a data register number 0-9.

6. Change the example to start with 15 and add the numbers back to 1. Changes?

   -- Store 15 in register 00

   -- 15 STO 02

   -- $^{*}\boxed{\text{Op}}$ 32

## 5-5 NESTED LOOP

- One loop inside another loop
- Flow diagram
- Example will follow the indirect instruction

## 5-6 INDIRECT INSTRUCTIONS

1. Key sequence -- Instruction Ind nn

   Where: Instruction = Store, recall, sum or any instruction

   Ind = indirect command

   nn = a data register number

2. Basic concept -- The indirect instruction is a two stage inquiry of data register memory. The command goes to a data register to find the data register number where information is to be found. It's just like telling someone to "go ask Sam where Fred is" instead of telling the person to "go find Fred."

3. Refer to p. V-68 of Personal Programming for a complete list of indirect addresses.

## 5-7 LOOP USING INDIRECT COMMAND TO RECALL DATA

1. Problem -- A person has five fields growing five different crops. Calculate the cash value of the crops given the acres in each field, the yield per acre of each field and the price per unit for each crop.

2. Calculation equation

$$\text{Cash receipts} = \sum_{i=1}^{5} A_i \times Y_i \times P_i$$

   where: A = acres

   Y = yield per acre

   P = price per unit

3. Write a program to calculate the cash receipts without using an indirect recall.

   a. Data register assignment

| Data registers | Factor | Fields |
|---|---|---|
| 18-22 | Acres | 1-5 |
| 24-28 | Yield | 1-5 |
| 30-34 | Price | 1-5 |

   b. Program

```
Lbl A
0 STO 06
RCL 18 x RCL 24 x RCL 30 = SUM 06
RCL 19 x RCL 25 x RCL 31 = SUM 06
RCL 20 x RCL 26 x RCL 32 = SUM 06
RCL 21 x RCL 27 x RCL 33 = SUM 06
RCL 22 x RCL 28 x RCL 34 = SUM 06
RCL 06
R/S
```

   c. 63 program steps

4. Write a program to calculate the cash receipts using an indirect recall

   a. Data register assignment -- same as previous

   b. Program

```
    Lbl A
    0 STO 06
    5 STO 04
    18 STO 00
```

```
24 STO 01
30 STO 02
Lbl lnx
RCL Ind 00 x RCL Ind 01 x RCL Ind 02 = SUM 06
Op 20  Op 21  Op 22
Dsz 4 lnx
RCL 06
R/S
```

c. 45 program steps

## 5-8 LOOP USING INDIRECT COMMAND TO STORE DATA

1. STO Ind nn: where nn is a register

2. Example -- Sum of years-digits depreciation

   5 years useful life

   $15,000 machine cost

   No salvage

   Sum of years = 15

3. Calculation procedure

| Year | Calculation |
|---|---|
| 1 | 5 ÷ 15 x 15,000 = 5,000 |
| 2 | 4 ÷ 15 x 15,000 = 4,000 |
| 3 | 3 ÷ 15 x 15,000 = 3,000 |
| 4 | 2 ÷ 15 x 15,000 = 2,000 |
| 5 | 1 ÷ 15 x 15,000 = 1,000 |

## 5-9 MULTIPLE TECHNIQUE PROBLEM

1. Problem -- A farmer grows three crops (corn, soybeans and wheat) in a number of fields, calculate the average yield, total production, total acres and cash receipts for each crop. Calculate the cash receipts and total acres for all the crops.

2. Techniques to use in the program --
   - Nested loop
   - Indirect command to store data
   - Indirect command to recall data
   - Increment operation code

## 5-10 MULTIPLE TECHNIQUE PROBLEM SOLUTION

1.

| Output Crop | Average Yield (Bu) | Total Production (Bu) | Acres | Cash Receipts |
|---|---|---|---|---|
| Corn | 97 | 15,035 | 155 | $39,842.75 |
| Soybeans | 48.7 | 3,650 | 75 | 22,082.50 |
| Wheat | 62 | 1,550 | 25 | 5,657.50 |
| Total | | | 255 | $67,582.75 |

2. Program — Comment

```
Lbl A
 0  STO 42    Total farm acres - Initialize
    STO 43    Total farm cash receipts - Initialize
10  STO 01    Acres input register number
17  STO 02    Yield input register number
24  STO 03    Price input register number
```

```
30 STO 04          Average yield output register number

33 STO 05          Total production output register number

36 STO 06          Acres output register number

39 STO 07          Cash receipts output register number

 3 STO 08          Set inside loop counter

   STO 09          Set outside loop counter

Lbl B              Ouside loop label

 0 STO Ind 06      Initialize acres register

   STO Ind 05      Initialize total production register

Lbl C              Inside loop label

RCL Ind 01
Sum Ind 06
Sum 42             Sum acres

RCL Ind 01 x
RCL Ind 02
= Sum Ind 05       Sum production

Op 21  Op 22       Increment counter registers

Dsz 8 C            Inside loop address

RCL Ind 05 ÷
RCL Ind 06
= STO Ind 04       Calculate Average Yield

RCL Ind 05 x
RCL Ind 03
= STO Ind 07
Sum 43             Sum cash receipts

Op 23  Op 24
Op 25  Op 26
Op 27              Increment counter register

 2 STO 08          Reset inside loop counter

Dsz 9 B            Outside loop address

R/S                Stop

Lbl D              Label to recall output to display

30 STO 00          Initialize output counter

14 STO 09          Set loop counter

Lbl Cos            Loop label
```

```
RCL Ind 00     Recall output
R/S            Stop
Op 20          Increment counter register
Dsz 9 Cos      Loop address
R/S            Stop
```

3. 125 program steps

## 5-11 FLAGS

1. A flag is a two way switch controlled from elsewhere in the program which can transfer execution to another segment or allow the program to continue with the next instruction.

2. Commands

- To set flag -- St flg y where y is the flag number 1-6. Flags 6-9 have a special purpose.

- To make the transfer after a flag has been set -- If flg y, n or xxx where y is the flag number and n is a label or xxx is a program location.

## 5-12 SUBROUTINE

1. SBR n or xxx where transfer is made to label n or program location xxx.

2. Defined -- A portion of a program written separate from the main program that can be called many times during execution of the main program.

3. Transfer command from main program $\boxed{\text{SBR}}$ n or xxx.

4. Return command back to main program $\boxed{\text{INV}}$ $\boxed{\text{SBR}}$ -- RETURN

## 5-13 SUBROUTINE EXAMPLE

1. Problem -- A person has three loans from a financial institution. Calculate the annual payments on each loan assuming an amortized loan.

2. Calculation --

$$A = \frac{r}{1 - \left(\frac{1}{1+r}\right)^n}$$

Where:

A = Annual payment per $1 per loan

r = Annual interest rate expressed as a percent

n = number of annual payments

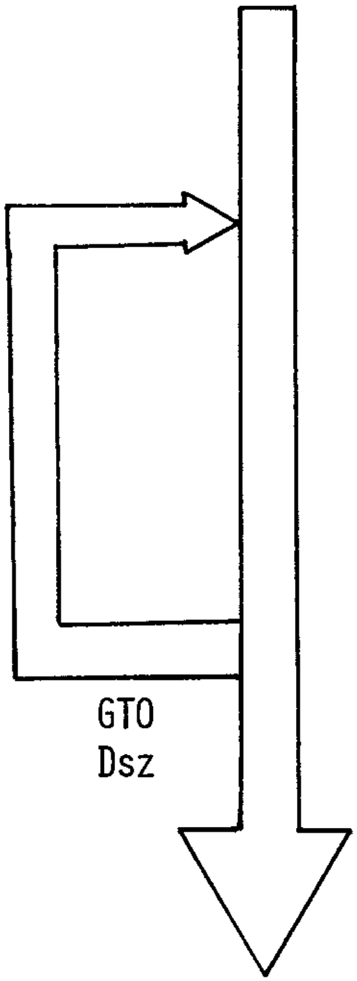
3. Data and data register assignment

| Loan | Data | Data Register | Comment |
|---|---|---|---|
| 1 | 100,000 | 15 | Outstanding balance |
| | 8 | 16 | Interest rate |
| | 15 | 17 | Years to repay |
| 2 | 25,000 | 18 | Outstanding balance |
| | 15 | 19 | Interest rate |
| | 7 | 20 | Year to repay |
| 3 | 10,000 | 21 | Outstanding balance |
| | 7 | 22 | Interest rate |
| | 5 | 23 | Years to repay |

4. Press B -- RCL 40, RCL 41, RCL 42

## PROGRAM EXECUTION FLOW

## LOOPING



# LOOPING WITH CONDITION TRANSFER

1. Dsz = DECREMENT, SKIP ON ZERO

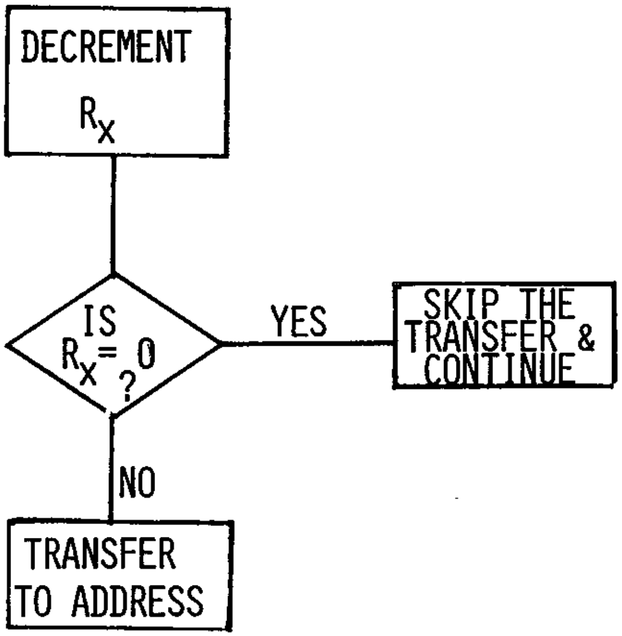
2. STATEMENT -- Dsz X, N OR NNN

   WHEN X = DATA REGISTER 0-9

   N = LABEL

   NNN = PROGRAM LOCATION

3. FLOW DIAGRAM



# LOOP EXAMPLE

SUM THE NUMBER 1 TO 5

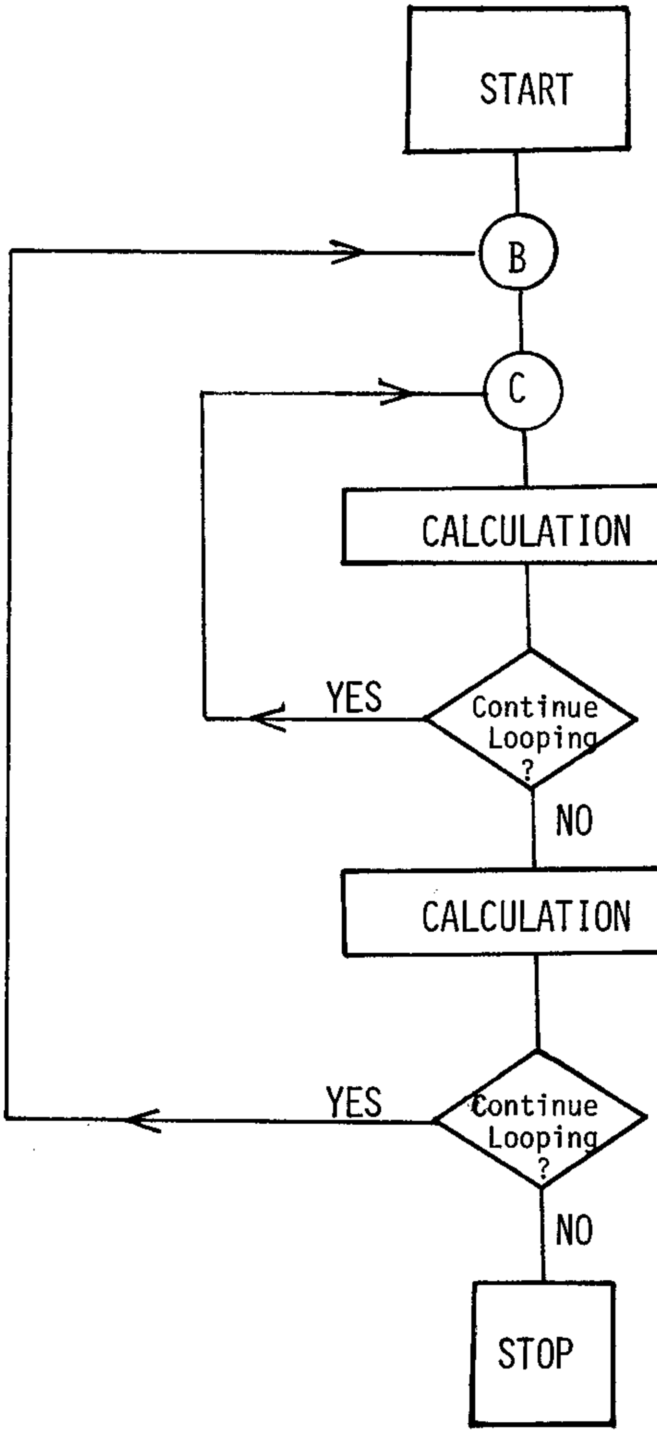
```
LBL A
5 STO 00
0 STO 02
R/S
LBL B
RCL 00
SUM 02
PAUSE
DSZ 0 B
RCL 02
R/S
```

# LOOP USING OPERATION CODES

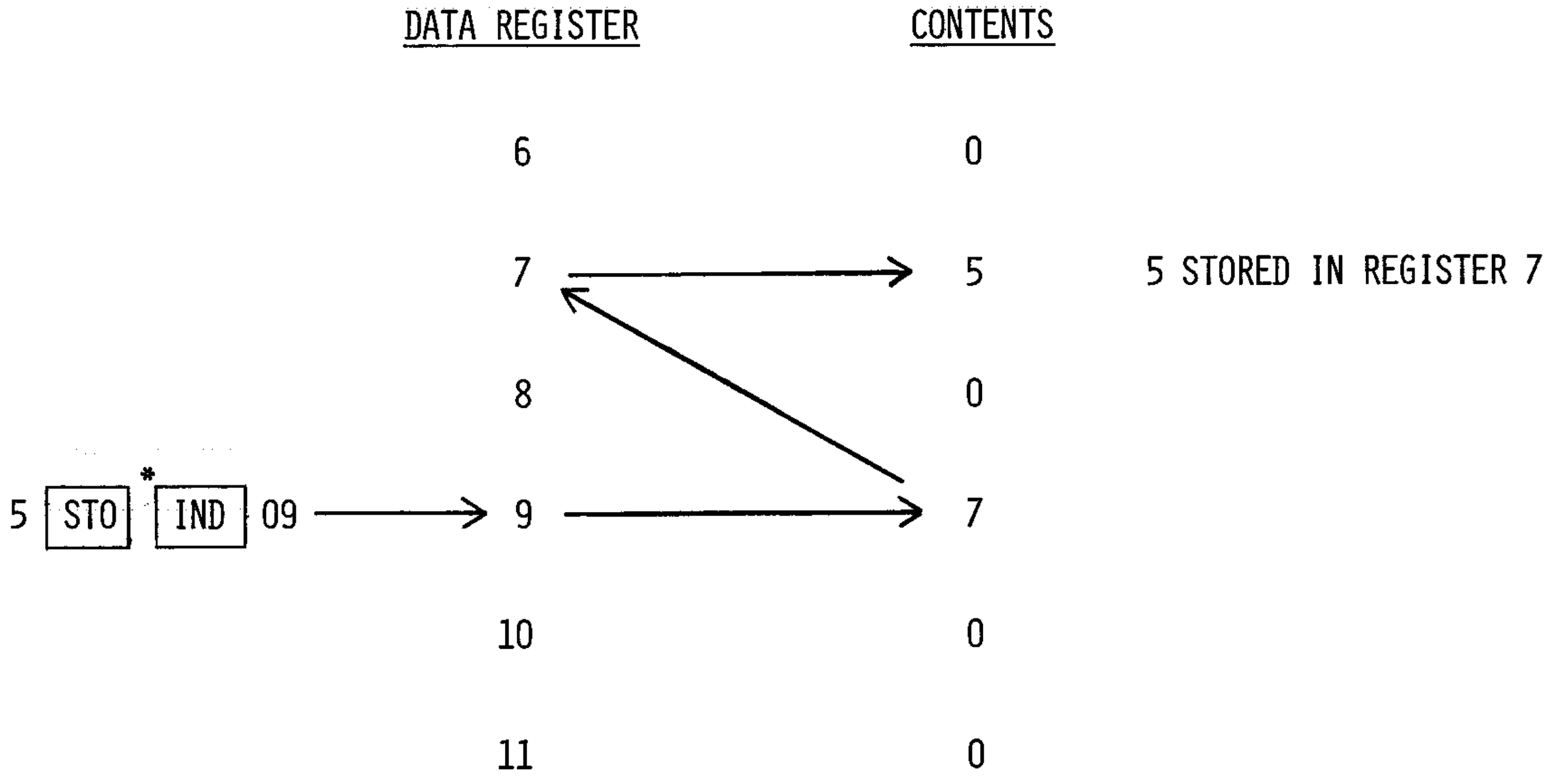
1. INCREMENT -- Op 20-29
   - *[Op] 2 N
2. DECREMENT -- Op 30-39
   - *[Op] 3 N
3. EXAMPLE -- ADD THE NUMBERS 1 THROUGH 10
   - STORE 10 IN REGISTER 00
   - PROGRAM

```
LBL A
1 STO 02
0 STO 21
LBL B
RCL 02  SUM 21
Op 22
Dsz 0 B
RCL 21
R/S
```

FLOW DIAGRAM OF CONCEPTUAL NESTED LOOP



# INDIRECT INSTRUCTIONS



DATA REGISTER CONTENTS

6 0

7 5 5 STORED IN REGISTER 7

8 0

5 STO * IND 09 9 7

10 0

11 0

# LOOP USING INDIRECT COMMAND TO RECALL DATA

(PROBLEM)

| FIELD | ACRES | YIELD PER ACRE | PRICE PER UNIT | CASH RECEIPTS |
|---|---|---|---|---|
| 1 | 32 | 90 BU. | $ 3.10 | $ 8,928 |
| 2 | 10 | 40 BU. | 3.80 | 1,520 |
| 3 | 58 | 85 BU. | 1.20 | 5,916 |
| 4 | 72 | 12 CWT | 22.00 | 19,008 |
| 5 | 43 | 20 TON | 33.00 | 28,380 |
| | | TOTAL CASH RECEIPTS | | $63,752 |

# LOOP USING INDIRECT COMMAND TO STORE DATA

## STORE DATA

```
    5 STO 00
    15000 STO 01
```

## PROGRAM

```
    Lbl A
    RCL 00 STO 02
    0 STO 10
    Lbl B
    RCL 02 SUM 10
    Dsz 2 B
    20 STO 03
    Lbl C
    RCL 00 ÷ RCL 10 x RCL 01 = STO Ind 03
    1 SUM 03
    Dsz 0C
    R/S
```

## ANSWERS

```
    RCL  20, 21, ETC.
```

## MULTIPLE TECHNIQUE PROBLEM

| CROP | DATA | DATA REGISTER | FIELD | FACTOR |
|---|---|---|---|---|
| CORN | 35 | 10 | 1 | ACRES |
| | 75 | 11 | 2 | ACRES |
| | 45 | 12 | 3 | ACRES |
| | 115 | 17 | 1 | YIELD |
| | 85 | 18 | 2 | YIELD |
| | 103 | 19 | 3 | YIELD |
| SOYBEANS | 55 | 13 | 1 | ACRES |
| | 20 | 14 | 2 | ACRES |
| | 50 | 20 | 1 | YIELD |
| | 45 | 21 | 2 | YIELD |
| WHEAT | 15 | 15 | 1 | ACRES |
| | 10 | 16 | 2 | ACRES |
| | 60 | 22 | 1 | YIELD |
| | 65 | 23 | 2 | YIELD |
| CORN | 2.65 | 24 | | PRICE PER BU. |
| SOYBEANS | 6.05 | 25 | | PRICE PER BU. |
| WHEAT | 3.65 | 26 | | PRICE PER BU. |

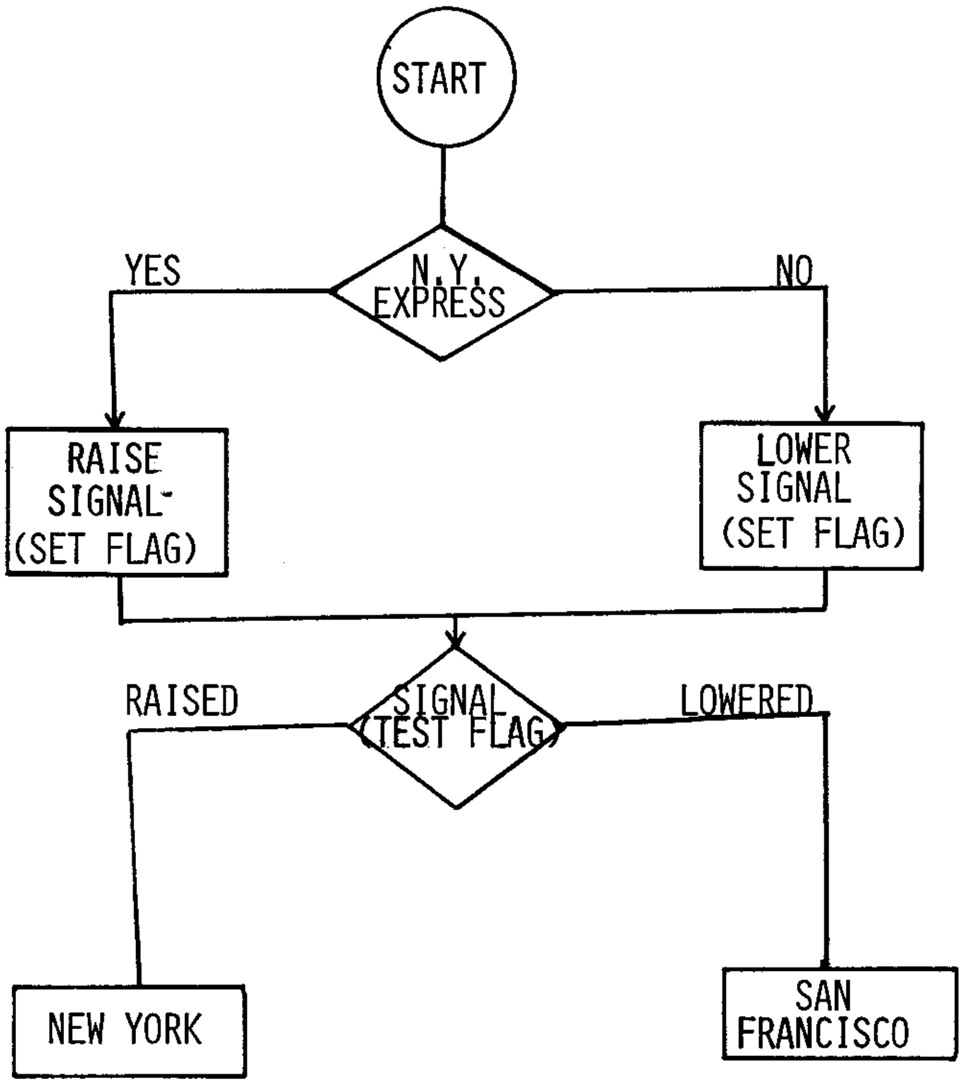
Data

```
   35.      10
   75.      11
   45.      12
   55.      13
   20.      14
   15.      15
   10.      16
  115.      17
   85.      18
  103.      19
   50.      20
   45.      21
   60.      22
   65.      23
   2.65     24
   6.05     25
   3.65     26
```

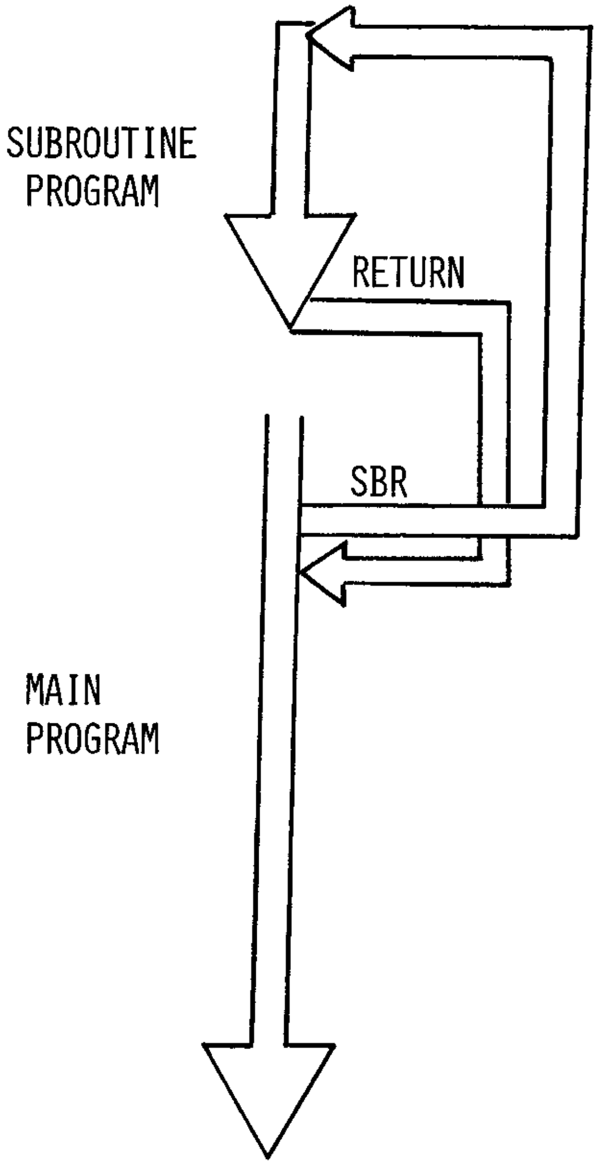
Program List

```
000  76  LBL
001  11   A
002  00   0
003  42  STO
004  42   42
005  42  STO
006  43   43
007  01   1
008  00   0
009  42  STO
010  01   01
011  01   1
012  07   7
013  42  STO
014  02   02
015  02   2
016  04   4
017  42  STO
018  03   03
019  03   3
020  00   0
021  42  STO
022  04   04
023  03   3
024  03   3
025  42  STO
026  05   05
027  03   3
028  06   6
029  42  STO
030  06   06
031  03   3
032  09   9
033  42  STO
034  07   07
035  03   3
036  42  STO
037  08   08
038  42  STO
039  09   09
040  76  LBL
041  12   B
042  00   0
043  72  ST*
044  06   06
045  72  ST*
046  05   05
047  76  LBL
048  13   C
049  73  RC*
050  01   01
051  74  SM*
052  06   06
053  44  SUM
054  42   42
055  73  RC*
056  01   01
057  65   ×
058  [illegible]  RC*
[illegible]
[illegible]
[illegible]
[illegible]  [illegible]  03
[illegible]  [illegible]  OP
[illegible]  21   21
065  69  OP
066  22   22
067  97  DSZ
068  08   08
069  13   C
070  73  RC*
071  05   05
072  55   ÷
073  73  RC*
074  06   06
075  95   =
076  72  ST*
077  04   04
078  73  RC*
079  05   05
080  65   ×
081  73  RC*
082  03   03
083  95   =
084  72  ST*
085  07   07
086  44  SUM
087  43   43
088  69  OP
089  23   23
090  69  OP
091  24   24
092  69  OP
093  25   25
094  69  OP
095  26   26
096  69  OP
097  27   27
098  02   2
099  42  STO
100  08   08
101  97  DSZ
102  09   09
103  12   B
104  91  R/S
105  76  LBL
106  14   D
107  03   3
108  00   0
109  42  STO
110  00   00
111  01   1
112  04   4
113  42  STO
114  09   09
115  76  LBL
116  39  COS
117  73  RC*
118  00   00
119  91  R/S
120  69  OP
121  20   20
122  97  DSZ
123  09   09
124  39  COS
125  91  R/S
```

# FLAGS


START
N.Y.
EXPRESS
YES
NO
RAISE
SIGNAL
(SET FLAG)
LOWER
SIGNAL
(SET FLAG)
SIGNAL
(TEST FLAG)
RAISED
LOWERED
NEW YORK
SAN
FRANCISCO

# SUBROUTINE



## SUBROUTINE EXAMPLE

```
LBL A
RCL 10 x .01 = STO 00
÷ (1 − (1 ÷ (1 + RCL 00)) Y^X RCL 11 = STO 02
INV SBR
LBL B
RCL 16 STO 10
RCL 17 STO 11
SBR A
RCL 02 x RCL 15 = STO 40
RCL 19 STO 10
RCL 20 STO 11
SBR A
RCL 02 x RCL 18 = STO 41
RCL 22 STO 10
RCL 23 STO 11
SBR A
RCL 02 x RCL 21 = STO 42
R/S
```

## LESSON 6 INTERNAL ROUTINES AND LIBRARY MODULES

Objectives

-- To describe the application of internal routines for calculator control operations and calculations

-- To apply internal routines in problem solutions

-- To learn the process for accessing and using solid-state module libraries

### 6-1 INTERNAL ROUTINES

1. Defined -- A subroutine or calculation sequence contained within the electronic circuitry of the calculator.

2. Types
   - Function keys (ie: $y^x$, $x^2$)
   - Special control operations -- See p. V-27 of Personal Programming for a complete list of all Op codes
   - Conversions -- p. V 30-32
   - Statistics -- p. V 32-40

### 6-2 STATISTICAL MEASURES -- SINGLE VARIABLE

1. Data

| Year | U. S. Corn Production (Million Bu.) |
|---|---|
| 75-76 | 5,829 |
| 76-77 | 6,266 |
| 77-78 | 6,505 |
| 78-79 | 7,268 |
| 79-80 | 7,939 |
| 80-81 | 6,645 |
| 81-82 | 8,201 |

2. Statistical Measures

$x$ = Yearly corn production

$\bar{x}$ = Mean corn production

$\sigma^2$ = Variance in corn production

$\sigma$ = Standard deviation of corn production

3. Steps to obtain the statistical measures

- Step 1 -- Clear data register 01 to 06
- Step 2 -- Enter data
- Step 3 -- Compute the mean -- 6950
- Step 4 -- Compute the variance -- 665,684
- Step 5 -- Compute the standard deviation -- 815

4. Error correction

- Error in number before pressing $^{*}\boxed{\Sigma +}$ Press $\boxed{\text{CE}}$, enter correct number and continue with normal entry.
- Error after pressing $^{*}\boxed{\Sigma +}$ Key in the incorrect number; press $\boxed{\text{INV}}$ $^{*}\boxed{\Sigma +}$ ; and reenter the number with normal entry.

## 6-3 STATISTICAL MEASURES -- TWO VARIABLES

1. Data

| Year | U.S. Corn Production (Million Bu.) | U.S. Price Per Bu. |
|---|---|---|
| 75-76 | 5,829 | 2.58 |
| 76-77 | 6,266 | 2.15 |
| 77-78 | 6,505 | 2.02 |
| 78-79 | 7,268 | 2.25 |
| 79-80 | 7,939 | 2.52 |
| 80-81 | 6,645 | 3.11 |
| 81-82 | 8,201 | 2.45 |

2. Steps to obtain the statistical measures

   - Step 1 -- Clear registers 01 to 06 and the t register

   - Step 2 -- Enter data in x, y form as follows:

     Enter x value<br>
     Press $\boxed{x \leftrightarrows t}$

     Enter corresponding y value<br>
     Press *$\boxed{\Sigma +}$

   - Step 3 -- Compute the y mean and the x mean -- $2.44 and 6950

   - Step 4 -- Compute the variance for y and x -- $ .11 and 665,684

   - Step 5 -- Compute the standard deviation for y and x -- $ .33 and 815

3. Error correction: Original x value, $\boxed{x \leftrightarrows t}$

   Original y value, $\boxed{\text{INV}}$, *$\boxed{\Sigma +}$

   Correct x value, $\boxed{x \leftrightarrows t}$

   Correct y value, *$\boxed{\Sigma +}$

## 6-4 LINEAR REGRESSION -- TWO VARIABLES

Step 1 -- Clear all data registers and the t register

Step 2 -- Enter data using the two variable format -- example of nitrogen fertilizer and corn yields, "Effect of Nitrogen Fertilizer on Corn Yield," Extension Bulletin E-802, Cooperative Extension Service, Michigan State University, Feb. 1979.

Step 3 --Compute the intercept and slope

*$\boxed{\text{Op}}$ 12 intercept

$\boxed{x \leftrightarrows t}$ slope

$y = 53.89 + .43036\ x$

Step 4 -- Compute the correlation coefficient and the index of determination

*$\boxed{\text{Op}}$ 13 correlation coefficient R=.95

$\boxed{x^2}$ index of determination $R^2$=.90

Step 5 -- Compute the means, standard deviations and variances for the x and y data

| Measure | x | y |
|---|---|---|
| $\bar{x}$ | 60 | 80 |
| $\sigma^2$ | 1600 | 330 |
| $\sigma$ | 40 | 18 |

Step 6 -- Expect y given x as 200

y = 140

## 6-5 SOLID-STATE MODULE FEATURES

1. Pre-programmed routines readily accessible using keys $^{*}\boxed{\text{Pgm}}$ nn where nn is the two-digit number of the program.

2. Module contains up to 5000 program steps or 15-25 magnetic cards.

3. Module is executed directly without using program memory.

4. Read only memory -- Cannot change the program in the module, but can download the program in memory, list, execute and change. Key sequence: $^{*}\boxed{\text{Pgm}}$ nn $^{*}\boxed{\text{Op}}$ 09 where nn is the two-digit number of the program.

5. Customized modules can be factory ordered -- i.e.: Iowa State University Agricultural Module.

6. Incorporation of module program as a subroutine in a magnetic card external program.

## 6-6 MODULE PROGRAM SUBROUTINES

1. Module library programs can be used as part of a user developed program stored in program memory.

2. A module as a subroutine saves programming time and memory storage. (A library module as a subroutine does not enter calculator memory, but operates electronically.)

3. Access instructions in program memory. (Refer to module library manual for the program labels, number and specific access instructions.) In general, the access instructions are:

   a. *$\boxed{\text{Pgm}}$ mm, N where mm is the module program number and N is the user-defined label in the module program. Execution transfers back to program memory at the end of label N section.

   b. *$\boxed{\text{Pgm}}$ mm $\boxed{\text{SBR}}$ N where mm is the module program and N is the common label in the module program. Execution transfers back to program memory at the end of label N section.

## INTERNAL ROUTINES

- FUNCTION KEYS
- SPECIAL CONTROL OPERATIONS
- CONVERSIONS
- STATISTICS

## STATISTICAL MEASURES -- SINGLE VARIABLE

STEP 1 -- CLEAR DATA REGISTER 01 TO 06

STEP 2 -- ENTER DATA FOLLOWED BY *$\boxed{\Sigma+}$ AFTER EACH NUMBER

5829    7939
6266    6645
6505    8201
7268

STEP 3 -- COMPUTE THE MEAN -- PRESS *$\boxed{\bar{x}}$

STEP 4 -- COMPUTE THE VARIANCE -- PRESS $\boxed{\text{INV}}$ *$\boxed{\text{OP}}$ 11

STEP 5 -- COMPUTE THE STANDARD DEVIATION -- PRESS $\boxed{\sqrt{x}}$

## STATISTICAL MEASURES -- TWO VARIABLES

STEP 1 -- CLEAR DATA REGISTER 01 TO 06 AND THE T REGISTER

STEP 2 -- ENTER DATA x FOLLOWED BY $\boxed{x \leftrightarrows T}$ AND y FOLLOWED BY *$\boxed{\Sigma+}$ AFTER EACH NUMBER

| X | Y | X | Y |
|---|---|---|---|
| 5829 | 2.58 | 7939 | 2.52 |
| 6266 | 2.15 | 6645 | 3.11 |
| 6505 | 2.02 | 8201 | 2.45 |
| 7268 | 2.25 | | |

STEP 3 -- COMPUTE THE MEAN

*$\boxed{\bar{x}}$ y MEAN

$\boxed{x \leftrightarrows T}$ x MEAN

STEP 4 -- COMPUTE THE VARIANCE

$\boxed{\text{INV}}$ *$\boxed{\text{Op}}$ 11 VARIANCE OF y VALUES

$\boxed{x \leftrightarrows T}$ VARIANCE OF x VALUES

STEP 5 -- COMPUTE THE STANDARD DEVIATIONS

REPEAT STEP 4 FOLLOWED BY $\boxed{\sqrt{x}}$

## LINEAR REGRESSION -- TWO VARIABLES

STEP 1 -- CLEAR ALL DATA REGISTERS AND THE T REGISTER

STEP 2 -- ENTER DATA USING THE TWO-VARIABLE FORMAT

| X | Y | X | Y |
|---|---|---|---|
| 0 | 45 | 80 | 93 |
| 20 | 63 | 100 | 96 |
| 40 | 77 | 120 | 98 |
| 60 | 86 | | |

STEP 3 -- COMPUTE THE INTERCEPT AND SLOPE

*$\boxed{\text{OP}}$ 12 INTERCEPT

$\boxed{x \rightleftarrows t}$ SLOPE

STEP 4 -- COMPUTE THE CORRELATION COEFFICIENT AND THE INDEX OF DETERMINATION

*$\boxed{\text{OP}}$ 13 CORRELATION COEFFICIENT R

$\boxed{x^2}$ INDEX OF DETERMINATION $R^2$

STEP 5 -- COMPUTE THE MEANS, STANDARD DEVIATION AND VARIANCES FOR THE X AND Y DATA USING THE TWO-VARIABLE FORMAT

STEP 6 -- EXPECTED Y GIVEN X

X

*$\boxed{\text{OP}}$ 14 EXPECTED Y

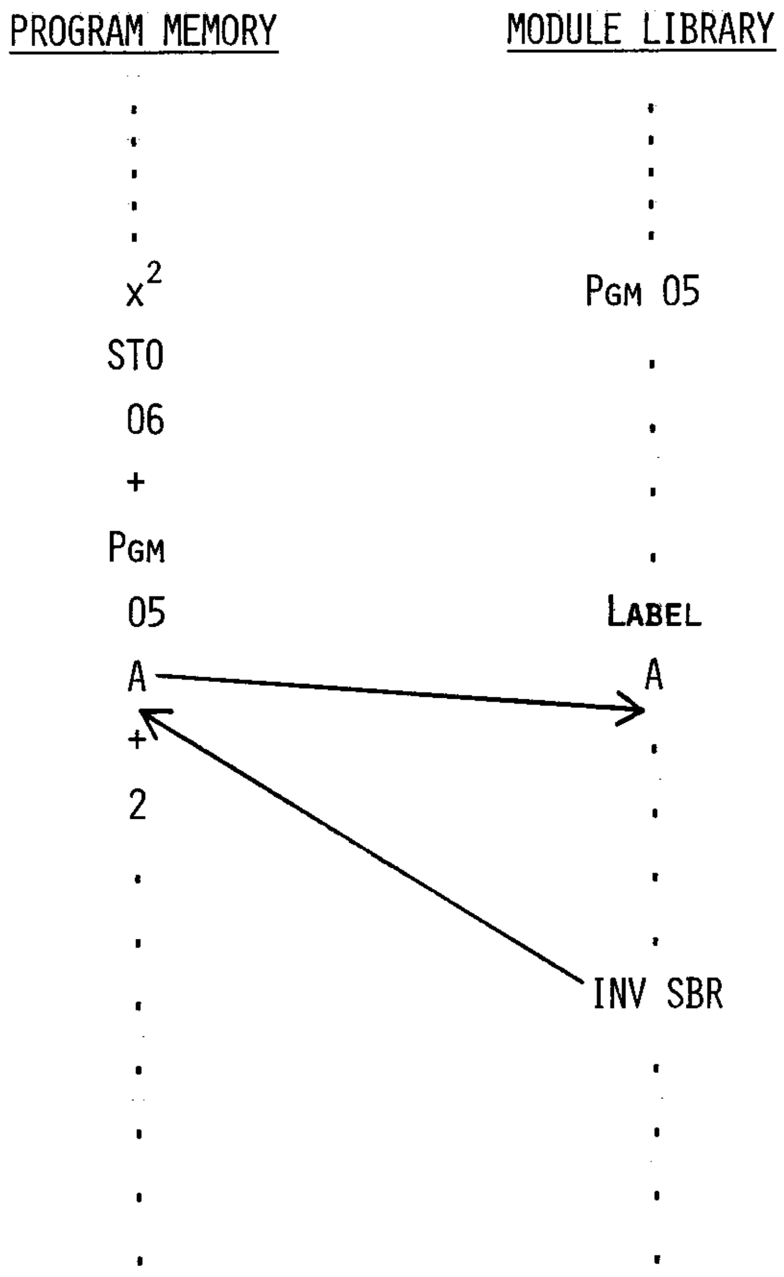
EXAMPLE PROGRAM -- COMPOUND INTEREST

| ENTER | PRESS | DISPLAY | COMMENTS |
|---|---|---|---|
| | *$\boxed{\text{Pgm}}$ 18 | 0 | SELECT PROGRAM |
| | *$\boxed{\text{E}'}$ | 0 | INITIALIZE |
| 1000 | $\boxed{\text{C}}$ | 1000.00 | PRESENT VALUE |
| 8 | $\boxed{\text{A}}$ | 8.00 | YEARS |
| 6 | $\boxed{\text{B}}$ | 6.00 | INTEREST |
| 0 | $\boxed{\text{D}}$ | 1593.85 | FUTURE VALUE |

DOWNLOAD PROGRAM

| ENTER | PRESS | DISPLAY | COMMENTS |
|---|---|---|---|
| | *$\boxed{\text{Pgm}}$ 18 | | SELECT PROGRAM |
| | *$\boxed{\text{Op}}$ 09 | | DOWNLOAD PROGRAM |
| | $\boxed{\text{RST}}$ | | RESTORE |
| | $\boxed{\text{LRN}}$ | | PROGRAM STEPS |

# MODULE PROGRAM SUBROUTINES



PROGRAM MEMORY

$x^2$
STO
06
+
PGM
05
A
+
2

MODULE LIBRARY

PGM 05
LABEL
A
INV SBR

## LESSON 7. USING THE PRINTER

Objectives

-- To understand the function of special control operations with the printer

-- To learn how to write programs to print alphanumeric characters

-- To be able to use the printer for plotting data graphically

### 7-1 CONTROL OPERATIONS FOR PRINTING

| Op Codes | Function |
|---|---|
| 00 | Initialize print register |
| 01 | 5 alphanumeric codes for print column |
| 02 | 5 alphanumeric codes for print column |
| 03 | 5 alphanumeric codes for print column |
| 04 | 5 alphanumeric codes for print column |
| 05 | Print contents of print register |
| 06 | Print last 4 characters of Op 04 with current display value |
| 07 | Plot a * in column 0-19 as specified by the display |
| 08 | List labels currently used in program memory |

### 7-2 CHARACTER POSITIONS

- Twenty characters can be printed on a line of paper output or a display value plus 4 alphanumeric characters

- The characters can be letters, numbers or symbols

- The 20 characters are divided into 4 groups with 5 alphanumeric characters per group. The four groups are positioned with an Op code as follows:

| | |
|---|---|
| Op 01 | Left 5 characters |
| Op 02 | Inside left 5 characters |
| Op 03 | Inside right 5 characters |
| Op 04 | Right 5 characters |

## 7-3 ALPHANUMERIC PRINTING ADDRESS CODE

- Each character is represented by a two digit code based on the row column position.

- Examples

| Code | Character |
|---|---|
| 00 | Blank |
| 01 | 0 |
| 13 | A |
| 47 | + |

## 7-4 ALPHANUMERIC PRINT EXAMPLE

1. Code alphanumeric messages

2. Write the program to print messages

3. Print the messages

   a. Clear fix-decimal, engineering notation and scientific notation

b. Clear the display register

c. Move code to display

d. Execute the Op 05 print command

4. Continue with program execution

## 7-5 SPECIAL PURPOSE CONTROL PRINT

- Print the value currently in the display plus
- Print the far right 4 characters on the same line
- Example

Printer line 43852 INC

## 7-6 PLOTTING DATA

1. Control operation Op 07 plots a * for current display value in character position 0-19 on the printer paper.

2. Example program using plotting for the program on p. 156, Programmable Calculators -- Business Application.

3. Data -- Net Farm Income

| Year | Grain Farm | Dairy Farm |
|---|---|---|
| 73 | $45,124 | $41,982 |
| 74 | 31,465 | 30,628 |
| 75 | 24,972 | 21,676 |
| 76 | 10,020 | 28,395 |
| 77 | 10,434 | 24,904 |
| 78 | 16,523 | 52,418 |
| 79 | 34,336 | 59,027 |
| 80 | 37,949 | 53,513 |

## CONTROL OPERATIONS FOR PRINTING

| | | |
|---|---|---|
| *OP | 00-06 | ALPHANUMERIC PRINTING |
| *OP | 07 | PLOTTING DATA |
| *OP | 08 | LIST PROGRAM LABELS USED |

CHARACTER POSITIONS

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Op 01 | | | | | Op 02 | | | | | Op 03 | | | | | Op 04 | | | | |

OR

(DISPLAY NUMBER)

| 16 | 17 | 18 | 19 |
|---|---|---|---|
| Op 06 | | | |

# ALPHANUMERIC PRINTING ADDRESS CODE

| | | UNITS DIGIT | | | | | | |
|---|---|---|---|---|---|---|---|---|
| TENS DIGIT | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 0 | blank | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 7 | 8 | 9 | A | B | C | D | E |
| 2 | - | F | G | H | I | J | K | L |
| 3 | M | N | O | P | Q | R | S | T |
| 4 | . | U | V | W | X | Y | Z | + |
| 5 | × | * | √ | π | e | ( | ) | , |
| 6 | ↑ | % | ! | / | = | ' | × | $\bar{x}$ |
| 7 | ≥ | ? | ÷ | ! | Π | ∴ | π | Σ |

# ALPHANUMERIC PRINT EXAMPLE

| SYMBOL | T | O | T | A | L | | I | N | C | O | M | E |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| CODE | 37 | 32 | 37 | 13 | 27 | 00 | 24 | 31 | 15 | 32 | 30 | 17 |

| PROGRAM | COMMENT |
|---|---|
| $\boxed{\text{CLR}}$ * $\boxed{\text{OP}}$ 00 | CLEARS PRINT REGISTER |
| 37 32 37 13 27 * $\boxed{\text{OP}}$ 01 | STORE "TOTAL" IN PRINT REGISTER |
| 00 24 31 15 32 * $\boxed{\text{OP}}$ 02 | STORE "(B) INCO" IN PRINT REGISTER |
| 30 17 00 00 00 * $\boxed{\text{OP}}$ 03 | STORE "ME (B)(B)(B)" IN PRINT REGISTER |
| * $\boxed{\text{OP}}$ 05 | PRINT COMPLETE TITLE ON PAPER |

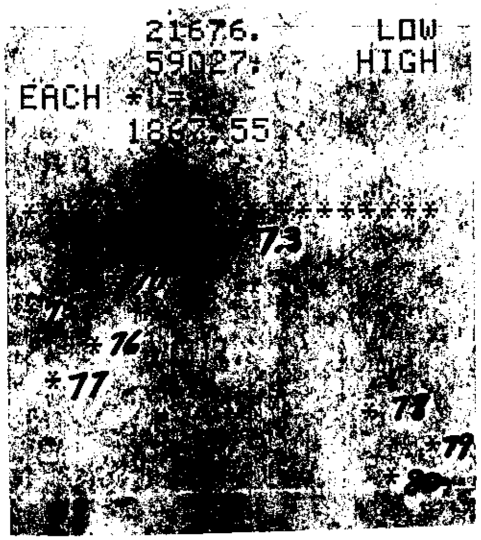
## SPECIAL PURPOSE CONTROL PRINT

```
      PROGRAM                      EXECUTION

         .
         .
         .
         .

  00 00 24 31 15                   XXXXX INC

       OP 04

       RCL 30

       OP 06
         .
         .
         .
         .
```

# PLOTTING DATA

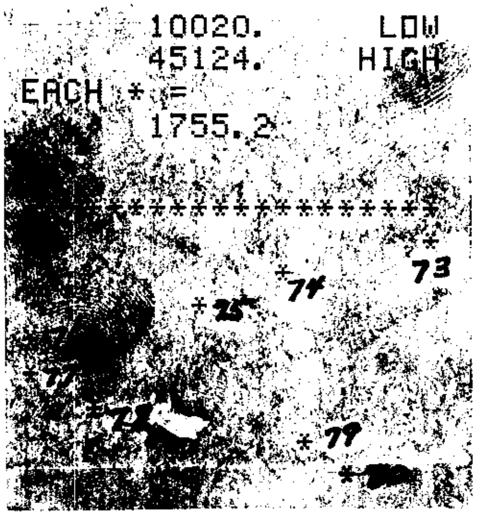
NET FARM INCOME

DAIRY FARMS



NET FARM INCOME

GRAIN FARMS



## LESSON 8. APPLICATION PROGRAMS

Objectives

-- To become aware of sources of library programs for the programmable calculator.

-- To practice operating programs prepared for the TI/59 programmable calculator

### 8-1 SOLID STATE SOFTWARE MODULES

Numerous modules can be purchased from TI retail outlets for TI programmable calculators on different topics. Each module contains 15-25 programs in a professional area.

### 8-2 PPX-59 PROFESSIONAL PROGRAM EXCHANGE

PPX
P.O. Box 53
Lubbock, TX 79408

A membership subscription to the PPX-59 software catalog containing a list of programs written by various users in many professional areas and a bimonthly newsletter. A subscriber can order programs for $4 per program.

Cost: $20 annual membership fee payable to PPX-59

### 8-3 TELCAL

Bulletin Office
Michigan State University
P.O. Box 231
East Lansing, MI 48824

Cost: $30 for a first-time subscriber and $10 annual renewal payable to Michigan State University.

Contents: A notebook with TI/59 programs in agriculture and forestry related areas.

8-4 PROGRAMS APPLIED TO AGRICULTURAL DECISIONS

Publications Distribution
Iowa State University
Ames, IA 50011

Cost: $30 or $1 per program payable to Iowa State University

Contents: A notebook with TI/59 programs in agriculture

8-5 AGRICULTURAL MODULE

ISU Research Foundation
315 Beardshear Hall
Iowa State University
Ames, IA 50011

Cost: $50 payable to ISU Research Foundation

Contents: Agricultural module and manual with 16 programs for the TI/58 and 59

8-6 NRAES-5 CALCULATOR PROGRAMS FOR EXTENSION

Northeast Regional Agricultural
Engineering Service
Riley-Robb Hall
Cornell University
Ithaca, NY 14853

Cost: $20 payable to NRAES

Contents: A notebook with TI/59 programs in agriculture

8-7 TI/59 PROGRAMMABLE CALCULATOR NOTEBOOK

Extension Agricultural Economics
Waters Hall
Kansas State University
Manhattan, KS 66506

Cost: $30 for a 3 year subscription payable to Kansas State University

Contents: A notebook with TI/59 programs in agriculture

8-8 HP-41C PROGRAMMABLE CALCULATOR PROGRAMS

Department of Agricultural Economics
Oregon State University
Corvallis, OR 97331

Contents: Several programs in agriculture for the HP-41C

8-9 TI/59 CALCULATOR PROGRAMS

Department of Agricultural Economics
Washington State University
Pullman, WA 99164

Contents: Several programs in agriculture for the TI/59

8-10 TI/59 CALCULATOR PROGRAMS

Department of Agricultural Economics
Texas A&M University
College Station, TX 77843

Contents: Several programs in agriculture for the TI/59

8-11 FOREST PROGRAMS FOR PROGRAMMABLE CALCULATORS E-1601

Bulletin Office
Michigan State University
P. O. Box 231
East Lansing, MI 48824

Cost: $1.10 payable to Michigan State University

Contents: A catalog listing of forestry programs for programmable calculators from various authors.

## MSU INTERNATIONAL DEVELOPMENT PAPERS

| | | Price |
|---|---|---|
| IDP No. 1 | Carl K. Eicher and Doyle C. Baker, "Research on Agricultural Development in Sub-Saharan Africa: A Critical Survey," 1982, (346 pp.). | $8.00 |
| IDP No. 2 | Eric W. Crawford, "A Simulation Study of Constraints on Traditional Farming Systems in Northern Nigeria," 1982, (136 pp.). | $5.00 |
| IDP No. 3 | M. P. Collinson, "Farming Systems Research in Eastern Africa: The Experience of CIMMYT and Some National Agricultural Research Services, 1976-81," 1982, (67 pp.). | $4.00 |
| IDP No. 4 | Vincent Barrett, Gregory Lassiter, David Wilcock, Doyle Baker, and Eric Crawford, "Animal Traction in Eastern Upper Volta: A Technical, Economic, and Institutional Analysis," 1982, (132 pp.). | $5.00 |
| IDP No. 5 | John Strauss, "Socio-Economic Determinants of Food Consumption and Production in Rural Sierra Leone: Application of an Agricultural Household Model with Several Commodities," 1983, (91 pp.). | $5.00 |

## MSU INTERNATIONAL DEVELOPMENT WORKING PAPERS

| | | |
|---|---|---|
| WP No. 1 | Daniel Galt, Alvaro Diaz, Mario Contreras, Frank Peairs, Joshua Posner, and Franklin Rosales, "Farming Systems Research (FSR) in Honduras, 1977-81: A Case Study," 1982, (48 pp.). | $0.00 |
| WP No. 2 | Edouard K. Tapsoba, "Credit Agricole et Credit Informel dans la Region Orientale de Haute-Volta: Analyse Economique, Performance Institutionnelle et Implications en Matiere de Politique de Developpement Agricole," 1982, (125 pp.). | $0.00 |
| WP No. 3 | W. P. Strassman, "Employment and Construction: Multi-Country Estimates of Costs and Substitution Elasticities for Small Dwellings," 1982, (48 pp.). | $0.00 |
| WP No. 4 | Donald C. Mead, "Sub-Contracting in Rural Areas of Thailand," 1982, (52 pp.). | $0.00 |
| WP No. 5 | Michael T. Weber, James Pease, Warren Vincent, Eric W. Crawford, and Thomas Stilwell, "Microcomputers and Programmable Calculators for Agricultural Research in Developing Countries," 1983, (113 pp.). | $5.00 |
| WP No. 6 | Thomas C. Stilwell, "Periodicals for Microcomputers: An Annotated Bibliography," 1983, (70 pp.). | $4.00 |
| WP No. 7 | W. Paul Strassman, "Employment and Housing in Lima Peru," 1983, (96 pp.). | $0.00 |
| WP No. 8 | Carl K. Eicher, "Faire Face a la Crise Alimentarie de l'Afrique," 1983, (29 pp.). | $0.00 |
| WP No. 9 | Thomas C. Stilwell, "Software Directories for Micro-Computers: An Annotated Bibliography," 1983 (14 pp.). | $3.00 |
| WP No. 10 | Ralph E. Hepp, "Instructional Aids for Teaching How to Use the TI-59 Programmable Calculator," 1983, (133 pp.). | $5.00 |

Copies may be obtained from: MSU International Development Papers, Department of Agricultural Economics, Agriculture Hall, Michigan State University, East Lansing, Michigan, 48824-1039, U.S.A. All orders must be prepaid in United States currency. Please do not send cash. Make checks or money orders payable to Michigan State University. There is a 10% discount on all orders of 10 or more copies. Individuals and institutions in the Third World may receive single copies free of charge.