SAMS 22414

TI-99/4A™ Calc

A Complete Electronic Spreadsheet

Gregory R. Schmalhofer

TI-99/4A™
CALC

Gregory R. Schmalhofer is a native of Lancaster, Pennsylvania and a graduate of Millersville State College (B. S. in computer science). Greg is currently employed by the Walter W. Moyer Company where he is a systems programmer. A microcomputer enthusiast since 1979, Greg uses both a TI-99/4A and a Commodore 64. In his spare time, he enjoys reading and playing basketball.

TI-99/4A™ CALC

by Gregory R. Schmalhofer

SAMS™
HOWARD W. SAMS & CO., INC.
4300 WEST 62nd STREET, P.O. BOX 7092
INDIANAPOLIS, INDIANA 46206

First Edition
First Printing—1984

International Standard Book Number: 0-672-22414-3
Library of Congress Catalog Card Number: 84-51273

Edited by *Douglas P. DeBrabant*
Product Development by *Troy Rondot*

Printed in the United States of America.

Preface

You are about to enter the world of electronic spreadsheets. Spreadsheet programs are probably the most widely used software products for microcomputers. They are popular for many reasons. Certainly because they are powerful but also because they have a wide range of applications, and are easy to use and understand.

One purpose of *TI-99/4A CALC* ("TI CALC") is to unleash the power of the TI-99/4A, without requiring you to purchase a lot of expensive hardware attachments. *TI CALC* was designed to be used on the base TI-99/4A home computer configuration: a TI-99/4A home computer, television, and a cassette recorder.

Home computing should be fun and enjoyable. *TI CALC* is that and more because it enables you to do any number of useful and exciting things. Happy computing!

GREGORY R. SCHMALHOFER

TI CALC is dedicated to my beautiful wife, Brenda, and my precious little girl, Emily. Together they fill my life with love and joy.

A NOTE TO THE READER

The programs in this book were not written as applications software but as educational examples of what your personal computer can do. All of the programs have been tested and work on the machine configuration for which they were designed. The programs are unprotected. This means that you can modify them to better understand how they work or to fit a different machine configuration.

WHAT IS A COMBO PACK?

A Combo Pack, like this package, is a step beyond your average technical book. While most books give you programming examples through printed listings (which we do here), Combo Packs provide the book with the listings recorded on magnetic media, either disk, cassette tape, or both.

Every effort has been made to be clear, concise, and informative about how these programs and routines work. If you experience any difficulty with the software operations, the solution can be found in the book or in your computer manuals.

We are rather proud of the time and effort that went into preparing the Combo Pack. If you have purchased and have enjoyed using it, let us know your thoughts. Your comments will be valuable in preparing future Combo Packs.

LOADING INSTRUCTIONS

The cassette accompanying this Combo Pack contains the program listing printed in the book and six spreadsheets.

To load a cassette file from this tape, perform the following steps:

1. Put the cassette tape into the cassette recorder.

2. Position the tape at the beginning of the tape.

3. Type **OLD CS1**
   Press <ENTER>

This will cause *TI CALC* to load into the computer's memory. When the program is loaded, it is ready to be used as described in the book. The spreadsheets can only be loaded using the main program as explained later in the book. For more complete loading instructions, see Chapter 3.

The following list shows the listing names and tape counter positions for the contents of the cassette tape. These numbers are approximate and may vary from recorder to recorder. They should, however, help you locate the programs you're searching for.

Tape Directory

Program Name	Counter Location
TI-99/4A CALC	0
AUTO GAS MILEAGE	28
TEMPERATURE CONVERSION	43
PIGGY BANK COUNTER	63
INCOME PLANNING	89
LOAN ANALYSIS	111
SAVINGS PLAN	134

CONTENTS

CHAPTER ONE
Introduction

Welcome to the fascinating world of electronic spreadsheets! With your purchase of *TI CALC* you now have one of the best cassette-based spreadsheet programs available for the TI-99/4A home computer.

In the Beginning

In general, spreadsheet programs are a relatively new type of software. They are probably less than ten years old, although handwritten spreadsheets, tedious to prepare and revise, have been used in business for a long, long time. Way back when, when there were no microcomputers, there were no electronic spreadsheets.

The original Apple computer gave birth to the very first spreadsheet software. It was this Apple-generated spreadsheet program that eventually became VisiCalc. It is interesting to note in retrospect that from this very humble beginning spreadsheets are now available for virtually every microcomputer. Another indication of their popularity is that spreadsheets have now migrated from micros to mainframes and in so doing spreadsheets have been established as programs of tremendous utility. At the same time, microcomputers became firmly established as legitimate business computers.

The Ultimate Program

Unfortunately, there is no one program that will do all things for all people. Even though this program or that may be very useful, it can't do everything. *TI CALC* is certainly no exception.

However, *TI CALC* is an excellent general purpose development system. Many of the things that normally require BASIC programming can now be done by setting up a *TI CALC* spreadsheet. Applications that perhaps you would never think of programming can now be done at a level above ordinary programming. You don't have to concern yourself with the many considerations required to do BASIC programming. For example, you don't have to worry about the **OPEN** or the **DATA** statement. Most of the programming is taken care of by *TI CALC*.

Another reason *TI CALC* is so easy to use is that there are so few commands to worry about. Whereas TI BASIC has over 80 commands, *TI CALC* uses only eight. As a result, you can learn to use this program very quickly and even if after not using it awhile, you can refresh your memory in minutes.

Now that you know something of spreadsheets and something of the features of *TI CALC*, let's go on to Chapter 2 and put the power of the TI-99/4A to work for you.

CHAPTER TWO
TI CALC FUNDAMENTALS

What Is a Spreadsheet?

The *TI CALC* spreadsheet is made up of 26 rows and 26 columns. The point at which a row intersects with a column is called a cell. Because we have 26 rows and 26 columns that means we have a total of 676 cells. Each cell can contain up to 12 characters or 12 digits. The entire spreadsheet within *TI CALC* memory is configured as in Fig. 2-1.

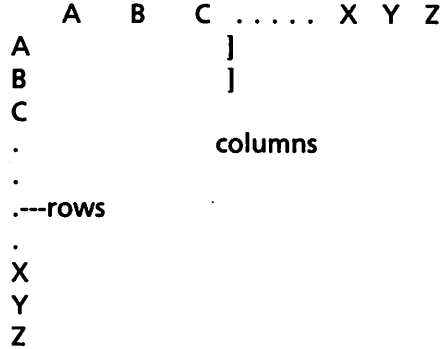

Fig. 2-1. The entire *TI CALC* spreadsheet.

This is exactly what you have available. However, it is not exactly what you see on the screen. Because the spreadsheet is so large, *TI CALC* can't display it all at once. It displays only a subset. A typical screen in *TI CALC* is shown in Fig. 2-2.

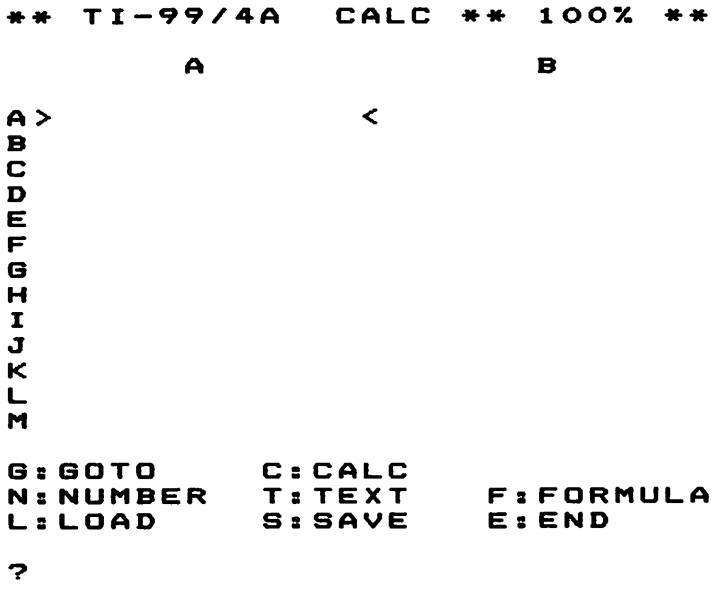

Fig. 2-2. A typical screen in *TI CALC*.

Even though we have many more rows and columns to work with, this is the subset that we typically see onscreen.

What Is a Cell?

We noted earlier that a cell is the *point* where a row and a column intersect. That is not quite true. Rather, a cell is the *area* where a row and column intersect. The proper way to refer to a cell, or a cell's area, is to use the letter designating the row first, followed by the letter designating the column. See Fig. 2-3 for examples showing cells AA and GB as illustrations of how cells have area. Remember that the space available for data is up to 12 characters or 12 digits long.

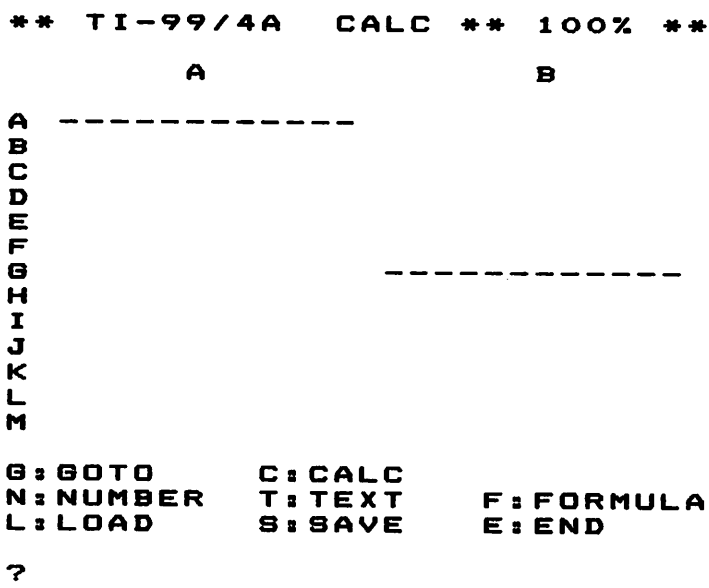

Fig. 2-3. Areas of cells AA and GB.

A cell's chief function is to hold data. What is input as data is up to you. You can input letters, numbers, and, eventually, formulas in the cell. The complete set of valid cell entries is the letters, numbers, and special symbols on your keyboard. These characters can be entered within the area of any cell.

How Cells Are Used

A cell can hold any one of three types of data. The three types of data or cell entry are text entry, number entry, and formula entry. Text entry means characters only. Number entry means, of course, numbers only. Formula entry means an arithmetic operation from one cell to another.

Text data is very versatile. It can be letters, numbers, letters and numbers, or any of the special symbols.

Numeric data is more specific than text (string) data. This type of cell can contain a valid number only. A valid number is defined as any positive or negative number.

All formulas must take the following form:

+ row column operation row column ...

To indicate a formula the first position must be a plus sign. Row and column are each one letter to determine the cell. Operation may be either addition subtraction, multiplication, division, or exponentiation, +, -, *, /, ^, respectively. Please note: there can be no spaces or numbers in a formula. Only the format shown above is correct. Some examples of the various cell types may help you understand their uses:

1. String entry examples
   a. *** name ***
   b. John Smith
   c. 05/25/84
2. Numeric entry examples
   a. 2000
   b. -2000.20
   c. 3.1427
3. Formula entry examples
   a. +BA+CA
   b. +DC-DD*DA
   c. +CC/CB

Screen Basics

Now that you know more about what a spreadsheet is and how cells make up spreadsheets, you will see right away that you need a way of moving to different cells of the spreadsheet. The methods of moving or changing what we see of the larger underlying spreadsheet are important if we are to become capable of taking full advantage of *TI CALC*.

When you start *TI CALC* the initial screen will appear as shown in Fig. 2-4.

```
** TI-99/4A   CALC ** 100% **

          A                B

A>                  <
B
C
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?
```

Fig. 2-4. *TI CALC* initial screen.

Notice that the cell at row A and column A (cell AA) has delimiters (the greater than and less than signs), one at each end of the cell. The area in between these delimiters is the area of cell AA that can hold data. The cell that is enclosed by these delimiters is known as the active cell. Only one cell at a time can be marked by delimiters, only one cell can be active at a time.

To move the active cell you should familiarize yourself with the cursor control keys. These are the keys on your TI-99/4A computer with the up, down, right, and left arrows on them. To use the cursor keys you must hold down the function key (labeled <FCTN>) while pressing the appropriate cursor arrow key. For example, to move the active cell to row C and column A (cell CA) hold down the <FCTN> key and press the down arrow key twice. See Fig. 2-5 for how the screen would look.

```
**  TI-99/4A   CALC  **  100%  **

          A                   B

A
B
C>                    <
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT       F:FORMULA
L:LOAD       S:SAVE       E:END

?
```

Fig. 2-5. Cell CA as the active cell.

The active cell is now cell CA, because it is the cell marked by delimiters.

In a similar fashion, if you wanted to move the active cell from cell CA to cell GB, you hold down the <FCTN> key while pressing the down arrow key four times and then press the right arrow key once. See Fig. 2-6 for how the screen would look.

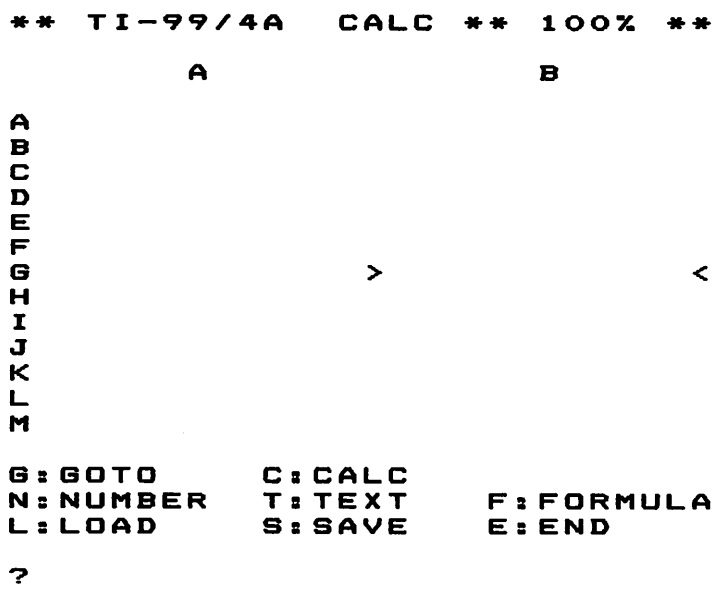

Fig. 2-6. Cell GB as the active cell.

Now that we can move the active cell around the screen readily, let's explore the concept of the active cell a little more. Notice the question mark below the letter M in the row identifiers. This question mark is the symbol for input. The bottom line where the question mark always appears is called the command line. The command line is in direct relationship with the active cell because it is through the command line that you input entries, or data, into the active cell. Basically, once you have positioned the active cell to the cell you want to put data in, or edit data in, you type what you want to enter into the active cell on the command line. Then you press the <ENTER> key and the data from the command line is moved to the active cell. To actually enter data, however, there is a little more you need to know—and that is discussed in Chapter 4.

So far we have discussed the subset of the entire spreadsheet. The subset is displayed when *TI CALC* is first begun. *TI CALC* has a facility to very quickly bring another column or group of rows onto the spreadsheet grid. Compare Figs. 2-7 and 2-8 to see how this facility works.

```
**  TI-99/4A   CALC  **  100%  **

          A                   B

A>                    <
B
C
D
E
F
G
H
I
J
K
L
M

G: GOTO       C: CALC
N: NUMBER     T: TEXT      F: FORMULA
L: LOAD       S: SAVE      E: END

?
```

Fig. 2-7. *TI CALC* initial screen.

The active cell is cell AA and the columns are A and B. If you want to see the next column, column C, hold down the <FCTN> key and press the right arrow key twice. When you press the right arrow key the first time, the active cell will be at cell AB. Or, on the rightmost side of the spreadsheet. By pressing the right arrow key again, *TI CALC* "realizes" that the next column is off the screen and it will then redisplay the spreadsheet with the next column, C, displayed. The screen would then appear as shown in Fig. 2-8.

```
** TI-99/4A   CALC ** 100% **

          B                C

A                  >                  <
B
C
D
E
F
G
H
I
J
K
L
M

G: GOTO       C: CALC
N: NUMBER     T: TEXT      F: FORMULA
L: LOAD       S: SAVE      E: END

?
```

Fig. 2-8. Column C displayed.

This same method can be used if you want to see the next row. For example, if the active cell is cell MC and you press the <FCTN> key and the down arrow key once the screen would appear as shown in Fig. 2-9.

```
** TI-99/4A   CALC ** 100% **
          B                C

B
C
D
E
F
G
H
I
J
K
L
M
N                  >                <

G: GOTO      C: CALC
N: NUMBER    T: TEXT      F: FORMULA
L: LOAD      S: SAVE      E: END

?
```

Fig.2-9. Row N displayed.

You should now be able to move the active cell around the screen with some confidence. You will probably discover that learning to move the active cell around on the spreadsheet quickly is a tremendous time saver.

Special Note

The number* in the upper right of the screen is the remaining percentage of free memory. *TI CALC* will permit you to continue entering data, but if memory becomes full, you will cause a serious error; you will also lose all variable amounts. You also won't be able to continue. Also, if you are using Extended BASIC you must change 5000 in line 130 of the program to 4200 and 50 in line 210 to 42. This is because TI Extended BASIC requires 800 bytes more memory than TI BASIC.

Well, now that you have some background in the intricacies of *TI CALC*, let's load the program and begin experimenting with some of the spreadsheet commands.

* Recalculated whenever the CALC routine is run or else whenever the entire screen is reprinted.

CHAPTER THREE
TI-99/4A Operating Procedures

The following instructions assume you have purchased the Combo Pack. If you have purchased the book only, you need to keyboard in the TI CALC program starting on page 77 before proceeding.

Setting Up the Cassette Recorder

TI CALC is encoded on cassette tape. You therefore need a cassette recorder, the only "extra" device needed to run the program. To be sure the cassette recorder is connected correctly, look over the following instructions for using the cassette interface cable.

1. Insert the single "plug" end of the interface cable (with the 9-pin connector) into the 9-pin outlet on the back right side of the TI-99/4A.
2. Attach the triple "plug" ends into the cassette recorder:
   a. Insert the plug with the red wire into the microphone jack.
   b. Insert the plug with the black wire into the remote jack.
   c. Insert the plug with the white wire into the earphone jack, or the external speaker jack.

If your cassette recorder does not have a remote jack, but does have the other two jacks, you can still use the recorder. What will happen is that the computer will not start and stop the cassette tape automatically. For example, normally when you are loading a tape and press **Play** on the cassette recorder the tape will not move right away, but rather it will start moving when the computer signals it to

begin playing. Without the remote jack, when you press **Play**, the tape will begin moving immediately. This should not present a problem especially if you leave plenty of room between tape files.

The cassette recorder should be positioned approximately two feet away from the television to avoid loading or saving problems. Set the volume on the cassette recorder to about the half way mark. Also, set the treble anywhere from the half way mark to the full mark. The cassette recorder should now be connected properly and ready to use.

What Is On the Tape?

The cassette tape has the same information on both sides. The *TI CALC* program is on Sides A and B. There are also six sample *TI CALC* spreadsheets, ready to load and use, duplicated on both sides of the tape. For a list of these spreadsheets and their tape locations see "A Note to the Reader" in the front of this book.

How to Load *TI CALC*

To run *TI CALC* or to use any of the stored spreadsheets you first must load the *TI CALC* program. Whenever you want to use *TI CALC*, apart from when you're starting the computer, loading the *TI CALC* program is the first thing to do. First you should turn the computer on and select option number "1" for TI BASIC. Once you receive the TI BASIC "READY" prompt you should get the cassette recorder and cassette tape ready for loading. Place the cassette with Side A facing up into the recorder and rewind the tape to the beginning. Type the command **OLD CS1** and press <ENTER>. TI BASIC will prompt you with the following message:

*** REWIND CASSETTE TAPE CS1**
THEN PRESS ENTER

Once you are sure the tape is rewound to the beginning of the tape press <ENTER>. TI BASIC will then prompt:

*** PRESS CASSETTE PLAY CS1**
THEN PRESS ENTER

Simply press **Play** on the recorder and then press <ENTER> on the computer. The computer will then begin loading the *TI CALC* program into memory and the following message will be displayed:

*** READING**

Once the program is loaded TI BASIC will then display:

*** DATA OK**

* **PRESS CASSETTE STOP CS1**
  THEN PRESS ENTER

Press the cassette **Stop** and then press the <ENTER> key on the computer. When you receive the "TI BASIC READY" prompt, the *TI CALC* program is ready to run. To run *TI CALC* type **RUN** and press <ENTER>.

Now that you know what's what with spreadsheets and *TI CALC* and you know how to load the program, let's move on to discuss the *TI CALC* commands in Chapter 4.

CHAPTER FOUR
TI CALC Commands

With *TI CALC* you can do a great many things: you can store words, numbers, or even calculate arithmetic expressions. *TI CALC* provides a set of commands that permit you to tell *TI CALC* what function it is that you would like to perform. This set of commands is not a long list of difficult to understand abbreviations. But rather, it is a very concise group of easy to understand words. The entire group totals only 8 commands. To make using the commands even easier, the commands are shown on the screen at all times.

Command Menu

The command menu is displayed below the rows and columns of the spreadsheet. Fig. 4-1 is an example of what the *TI CALC* start up screen, with the command menu, looks like.

```
**  TI-99/4A   CALC  **  100%  **

          A                  B

A>                    <
B
C
D
E
F
G
H
I
J
K
L
M

G:GOTO        C:CALC
N:NUMBER      T:TEXT        F:FORMULA
L:LOAD        S:SAVE        E:END

?
```

Fig. 4-1. *TI CALC* start up screen.

To select any of the above commands you simply press the letter of the key associated with that command.

Brief Descriptions

GOTO is used to display, or go to, different parts of the spreadsheet.
CALC is used when you want to calculate the values for all of the formulas.
NUMBER is one of the cell entry types. This command permits you to enter a number into the active cell.
TEXT is the second of the cell entry types. This command permits you to enter any letter, number, or any of the special character keys. **TEXT** will accept any keyboard character.
FORMULA is the third of the cell entry types. This command permits you to enter an arithmetic expression into the active cell.
LOAD permits you to recall, or load, a spreadsheet that was saved on cassette tape by the **SAVE** command.
SAVE provides a way for you to save the spreadsheet currently in memory. The results can then be used as input to the **LOAD** command.
END is the proper way to end the program. Its use ensures that the data is saved if needed.

The commands **NUMBER, TEXT,** and **FORMULA** all enter data into the active cell. The other commands will prompt you for any other information if needed.

The GOTO Command

Function

GOTO will display or go to a different group of rows and columns.

Operation

To perform the **GOTO** command, press the **G** key. *TI CALC* will then prompt you for the row and column that you want to be the upper left cell of the spreadsheet when it is displayed.

Input

For row and column prompts, you can enter the letter of any row and the letter of any column.

Example

In the following example (beginning with Fig. 4-2) we want to display at row **N** and column **D**, cell **ND**. To do so, press the **G** key. *TI CALC* then displays a prompt for the row as you see in Fig. 4-2.

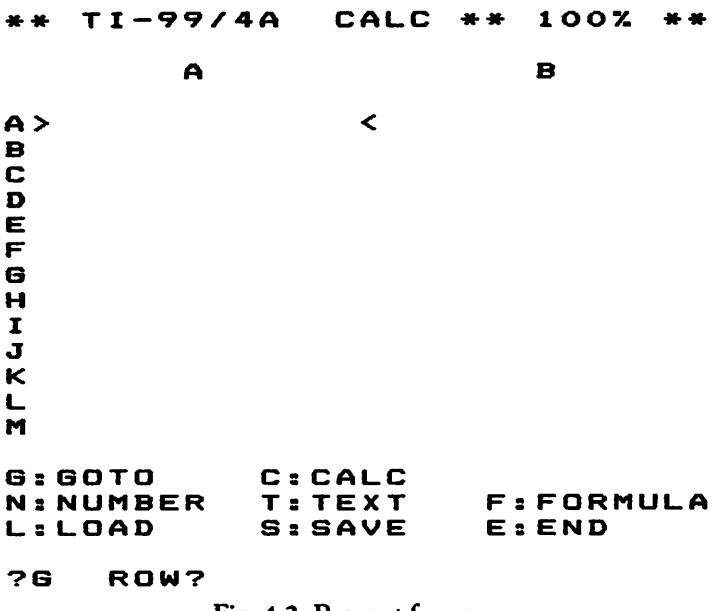

Fig. 4-2. Prompt for row.

To the prompt, type **N**. *TI CALC* then prompts for the column, for which you should type **D**. See Fig. 4-3.

```
**  TI-99/4A   CALC  **  100%  **

          A                   B

A>                    <
B
C
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT       F:FORMULA
L:LOAD       S:SAVE       E:END

?G   ROW? N     COLUMN? D
```

Fig. 4-3. Prompt for column.

Once the column is entered, *TI CALC* will then display the spreadsheet starting at the new row and column locations. See Fig. 4-4.

```
**  TI-99/4A   CALC  **  100%  **

          D                   E

N>                    <
O
P
Q
R
S
T
U
V
W
X
Y
Z

G:GOTO        C:CALC
N:NUMBER      T:TEXT        F:FORMULA
L:LOAD        S:SAVE        E:END

?
```

Fig. 4-4. The new row & column locations.

Note: the GOTO command is a fast way of moving to an area of the spreadsheet that is not currently displayed. You can also use the cursor arrow keys to automatically advance to the next row or column. However, it is usually best to use the cursor arrow keys to move around with the currently displayed spreadsheet and to use the GOTO command to display parts of the spreadsheet that are not currently displayed.

By staying in smaller areas you will get faster execution of the program and its routines. For example, a packed spreadsheet in rows A-N and columns A-D will execute much faster that one spread out over the entire spreadsheet.

The CALC Command

Function

CALC tells *TI CALC* to calculate any arithmetic expressions that are in the individual cells and then to display the results in the appropriate cell.

Operation

To use CALC press the C key. The CALC command does not prompt you for any other information.

Input

The CALC command requires no other information.

Example

In this example (beginning with Fig. 4-5), cell AA has the value of 500 and cell CA has the value of 200. Also in cell EA is the expression, +AA+CA, which was entered by the **FORMULA** command.

```
**  TI-99/4A   CALC  **  99%  **

          A                   B

A>          500.00<
B
C           200.00
D
E  +AA+CA
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?
```

Fig. 4-5. CALC example.

Now enter C and *TI CALC* immediately begins to go through the used portion of the spreadsheet and perform any existing calculations or formulas. See Fig. 4-6.

```
**  TI-99/4A   CALC  **  99%  **

          A                    B

A>          500.00<
B
C           200.00
D
E           700.00
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?
```

Fig. 4-6. Results of calculations.

Notes: The CALC command will only affect the cells that have formulas. If you have a spreadsheet that has a lot of formulas, CALC may take a few moments to complete, so it is best to make as many entries to the spreadsheet as possible before each recalculation. All calculations are performed even if the cell is not currently displayed.

You may have a calculation that takes a number from a cell that has another calculation. This is called embedding of calculations (chaining or linking of one calculation to another). You may embed as many calculations as needed. The only limit is the amount of memory in the computer.

When *TI CALC* begins to perform all of these calculations it does so in a predefined order or sequence. This is known as the order of calculation. Rows are primary. This means that the program performs all calculations in the first row (row A), then those in the second row (row B), and so on. Any cell used in a calculation which does not yet have a value will be defaulted to zero. You can reference any cell but if the cell involves a calculation, it must be referenced after the calculation or it will be defaulted to zero.

The NUMBER Command

Function

This command is used to enter a number or a numerical value into the active cell. The number entered is displayed with two positions to the right of the decimal and up to nine positions to the left. Numbers 10 or 11 digits long (without a decimal point) will be right justified without a decimal point.

Operation

To use the **NUMBER** command, press the N key. This command will then display a 12 position entry area on the command line for you to enter the number.

Input

This command will prompt you to enter the number in the entry area. Any positive or negative number can be entered. Should you need a number with more than nine positions to the left of the decimal you can use exponentiation. Exponentiation is indicated by the letter "E". The total set of valid characters is 0123456789.-E.

Example

This example (beginning with Fig. 4-7) has you enter the number 9.75 into cell CA. To do this, position the active cell to cell CA by using the cursor arrow keys. Then press the N key. At this point, the entry area will now be displayed between the two familiar delimiters. When you begin typing, the keystrokes will show in the entry area. *TI CALC* will allow only valid number keys to be entered. After you type in 9.75, the screen will look like that shown in Fig. 4-7.

```
**  TI-99/4A   CALC  **  100%  **

          A                   B

A
B
C>                    <
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?N    >9.75              <
```

Fig. 4-7. Using NUMBER command.

As soon as the <ENTER> key is pressed the number is moved from the entry area into the active cell. When <ENTER> is pressed the screen will look like Fig. 4-8.

```
**  TI-99/4A   CALC  **  99%  **

          A                   B

A
B
C>              9.75<
D
E
F
G
H
I
J
K
L
M

G: GOTO        C: CALC
N: NUMBER      T: TEXT        F: FORMULA
L: LOAD        S: SAVE        E: END

?
```

Fig. 4-8. Using NUMBER command, continued.

Note: The command line is then clear and ready for the next command. An exponential number is entered as 2.33E7.

WARNING: The only way to produce a program execution error and "bomb" out to intermediate level is to enter a number with an invalid numerical format. For example, a number with two decimal points or two E's. This error is not likely to occur but it's worth taking extra care to avoid it.

The TEXT Command

Function

This command is used to enter letters, letters and numbers, or any other keyboard character into the active cell.

Operation

To use the **TEXT** command press the **T** key. This command will then display a 12 position entry area on the command line for you to enter the string or text.

Input

This command will prompt you to enter the text in the entry area. Any combination of keyboard characters is considered a valid entry.

Example

This example (beginning with Fig. 4-9) has you enter the string DAD & MOM into cell CA. First, position the active cell to cell CA by using the cursor arrow keys. Then press T. The entry area of 12 positions is then displayed. After the text is entered in the entry area, the screen will look like that shown in Fig. 4-9.

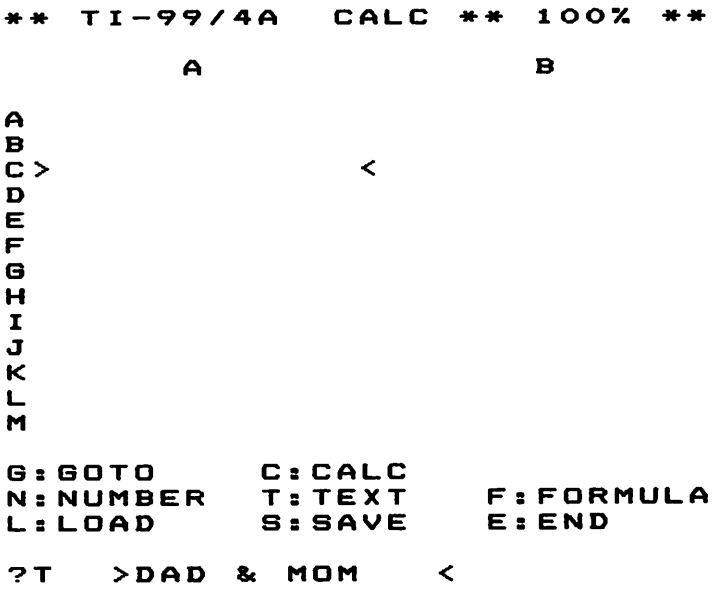

Fig. 4-9. Using TEXT command.

When <ENTER> is pressed the text is moved from the entry area into the active cell. Fig. 4-10 shows the results.

```
**  TI-99/4A   CALC  **  99%  **

          A                   B

A
B
C>DAD  &  MOM     <
D
E
F
G
H
I
J
K
L
M

G:GOTO        C:CALC
N:NUMBER      T:TEXT      F:FORMULA
L:LOAD        S:SAVE      E:END

?
```

Fig. 4-10. Using TEXT command, continued.

Note: The command line is then cleared for the next command. **TEXT** is very versatile as in the above example with the special symbol "&". Any combination of keyboard characters is possible. The string **June 16, 1984** is a valid **TEXT** entry. Also—a number can be entered as text but formulas will not "recognize" such input as a number and thus the cell's numerical value will be 0.

The FORMULA Command

Function

This command is used to enter an arithmetic expression into the active cell.

Operation

To use this command, press **F**. The program will then display the 12 position entry area for the calculation or formula to be typed in. A "+" is automatically displayed in the first position of the entry area. That way, the program can track the cells that are formulas to be calculated when **CALC** is used.

Input

The input for this command must be done by a strict format. The required format is: **+ row column operation row column. . . .**

Row and column may be any of the row and column identifiers A through Z. Operation can be any of the five operations: addition, subtraction, multiplication, division, or exponentiation.

Example

In this example (beginning with Fig. 4-11) the spreadsheet contains the number 25.00 in cell AA and the number 4.00 in cell BA. Let's suppose you want cell CA to equal cell AA times cell BA, or 4 times 25. First position the active cell to cell CA, then press **F**. The program will display the entry area with a "+" sign in the first position. Starting in the second position, the calculation would be entered as AA*BA. The screen would appear as you see it in Fig. 4-11.

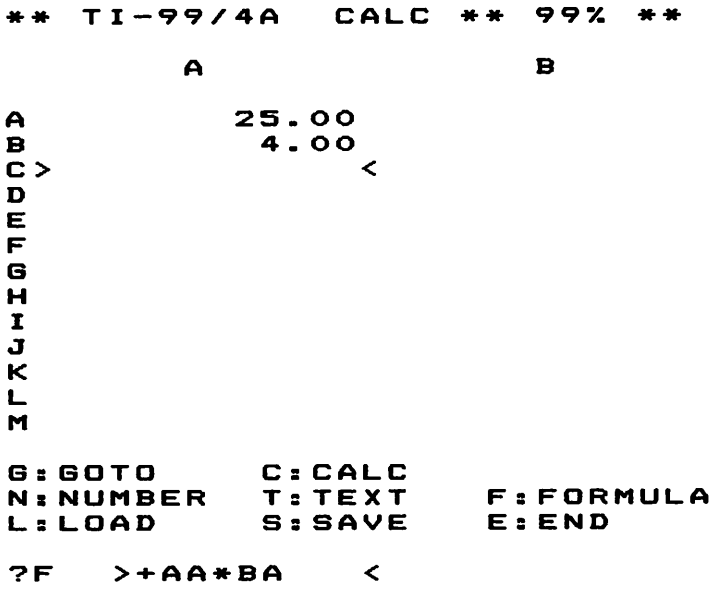

Fig. 4-11. Using FORMULA command.

When <ENTER> is pressed, the formula is displayed at the cell location. It will stay in the cell until the first time you perform a **CALC** command at which time it will be replaced with the results of the calculation. See Fig. 4-12.

```
** TI-99/4A   CALC ** 99% **

          A                    B

A              25.00
B               4.00
C>+AA*BA              <
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT       F:FORMULA
L:LOAD       S:SAVE       E:END

?
```

Fig. 4-12. Using FORMULA command, continued.

At this point, you could enter other commands but to see the results of what you've just done, perform CALC as described earlier. After the calculation is finished the screen will appear as you see it in Fig. 4-13.

```
**  TI-99/4A   CALC  **  99%  **

          A                   B

A              25.00
B               4.00
C>           100.00<
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?
```

Fig. 4-13. Using FORMULA command, continued.

Note: **FORMULA** also uses the entry area for input into the active cell, so the largest formula you can enter is up to four operations. For example, +AB+CD+EF+GH. You cannot embed numbers within the formula. For example, the following is an incorrect formula, +AB*99. To accomplish the same result, first enter the number 99 in a cell, then reference that cell in the calculation. For example, use **NUMBER** to put the number 99 into cell CB. Then enter **FORMULA** as +AB*CB. This will give you the desired results. The program will perform the operations in each formula as they are encountered within the expression.

To delete a formula totally, you must first use a **TEXT** edit to get rid of the initial "+", (that way you protect against formulas from being erased accidentally). Also, if you chose N and the active cell contains a formula, the program will force you to edit the formula and not the number in the cell.

The LOAD Command

Function

This command is used to load into memory a spreadsheet that was stored on cassette tape by using the **SAVE** command.

Operation

To use **LOAD**, press **L**. The program will first clear all values from the spreadsheet, clear the screen, and begin the loading process.

Input

You will receive a couple of instruction prompts for the cassette recorder. The prompts are described below:

1. **REWIND CASSETTE TAPE CS1**
   THEN PRESS ENTER
2. **PRESS CASSETTE PLAY CS1**
   THEN PRESS ENTER

Simply press <ENTER> when the appropriate cassette operation is complete.

Example

A typical loading process would look like the following:

CALC LOAD
* **REWIND CASSETTE TAPE CS1**
THEN PRESS ENTER
* **PRESS CASSETTE PLAY CS1**
THEN PRESS ENTER

When the spreadsheet is loaded correctly a prompt is given to stop the cassette recorder:

* **PRESS CASSETTE STOP CS1**
THEN PRESS ENTER

The spreadsheet is now loaded into the computer's memory and it is ready to use.

Note: Any spreadsheet that was stored by the program using **SAVE** can be loaded using **LOAD.**

The SAVE Command

Function

This command is used to store a *TI CALC* spreadsheet on cassette tape.

Operation

To use this command, press S. The program will clear the screen and begin the saving process.

Input

You will receive instruction prompts for the cassette recorder. The prompts are described below:

1. **REWIND CASSETTE TAPE CSI**
   THEN PRESS ENTER

2. PRESS CASSETTE RECORD CS1
   THEN PRESS ENTER

Press <ENTER> when the appropriate cassette operation is complete.

Example

A typical saving process would look like the following:

CALC SAVE
* REWIND CASSETTE TAPE CS1
  THEN PRESS ENTER
* PRESS CASSETTE RECORD CS1
  THEN PRESS ENTER

When the spreadsheet is completely stored, a prompt to stop the cassette recorder is given:

* PRESS CASSETTE STOP CS1
* THEN PRESS ENTER

The spreadsheet is now saved on cassette tape.

Note: A spreadsheet that was saved by the **SAVE** command can be loaded using **LOAD**.

The END Command

Function

This command ends the program and makes sure the spreadsheet is saved if needed.

Operation

To use this command, press **E**.

Input

The program will ask if you are sure you want to end. Type **Y** or **N**. If you type **N** you will return to ordinary command entry.

Example

When you enter the **END** command, the screen will look like you see it in Fig. 4-14.

```
** TI-99/4A   CALC ** 100% **

          A                   B

A>                    <
B
C
D
E
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT       F:FORMULA
L:LOAD       S:SAVE       E:END

?E   ARE YOU SURE (Y/N)?
```

Fig. 4-14. Using END command.

Notes: Although you can simply turn the computer off or press <FCTN/Q>, it is best to use the **END** command. That way you'll always be sure you saved the latest data.

CHAPTER FIVE
Creating a Spreadsheet

Chapter 5 brings all of the previous chapters together by actually going through the entire process of making a spreadsheet. While creating this sample spreadsheet you will be able to tell which features of *TI CALC* will be most helpful to you.

I am sure that everybody wonders from time to time what their car's gas mileage is. Well the sample spreadsheet that we will put together will let you answer that question.

Setting It Up

The first thing to do is to load the *TI CALC* program. If you need to refresh your memory of how to load *TI CALC* refer to Chapter 3. Once *TI CALC* is loaded type the BASIC **RUN** command. After a few moments the *TI CALC* screen will be displayed. At this point you are looking at a blank spreadsheet. When creating a spreadsheet you should try to picture what you want the screen to look like when you are finished. Try to plan ahead where you want any titles to be or where you want to put the calculations. Of course, if you try it one way and do not like the way it looks you can change it at any time.

The spreadsheet that you will create here will allow you to enter your beginning mileage, ending mileage, and the number of gallons

of gas used. When the calculation is completed, you will see your car's gas mileage.

Auto Gas Mileage

Note: type the spaces inside the quotes, but don't type the quotes.

Step #1

Press T (the **TEXT** command)
Type the words "**AUTO GAS**"
Press <ENTER>

Step #2

Press <FCTN/X> (while holding down the <FCTN> key, press X)
The active cell is BA.
Press **T**
Type "----------"
Press <ENTER>

Step #3

Press <FCTN/X> two times
The active cell is DA.
Press **T**
Type "**START MILES:**"
Press <ENTER>

Step #4

Press <FCTN/X>
The active cell is EA.
Press **T**
Type " **END MILES:**"
Press <ENTER>
The screen should appear as you see it in Fig. 5-1.

Step #5

Press <FCTN/X>
The active cell is FA.
Press **T**
Type "**GALLONS GAS:**"
Press <ENTER>

Step #6

Press <FCTN/X> two times
The active cell is HA.
Press **T**
Type " **MPG**"
Press <ENTER>

Step #7

Press <FCTN/D>
The active cell is HB.
Press **F** (the **FORMULA** command)
Type **+EB-DB/FB**
Press <ENTER>
This will subtract the starting miles (DB) from the ending miles (EB) and divide the result by the number of gallons used (FB). However, notice that at this point cells EB, DB and FB contain no figures. These three cells are the cells in which information will vary. They are your "what if" variables, because you can experiment by changing the numbers and then displaying the results.

Step #8

Press <FCTN/E> six times
The active cell is BB.
Press **T**
Type "----------"
Press <ENTER>

Step #9

Press <FCTN/E>
The active cell is AB.
Press **T**
Type "**MILEAGE**"
The screen will now appear as you see it in Fig. 5-2.

Step #10

Press **S** (the **SAVE** command)
This will save your work up to this point. For additional information on this command, see CHAPTER 4.

Steps #11-13

Now use <FCTN>, arrow keys, and the **NUMBER** command (N) to put numbers in cells DB, EB, and FB.

Step #14

The spreadsheet is now all set and ready for the first **CALC** command.
The active cell may be any cell.
Press C
Wait for the results of this command. The results will appear in cell HB. When the calculations are completed the screen will appear as in Fig. 5-3.

Step #15

After you are satisfied with the way the spreadsheet looks and works you can save the spreadsheet using the **SAVE** command (S). At this point, it would be good practice to save the spreadsheet and then load it back in to make sure everything works the way it should.

```
** TI-99/4A   CALC ** 98% **

          A                    B

A       AUTO GAS
B  ------------
C
D  START MILES:
E>   END MILES:<
F
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT      F:FORMULA
L:LOAD       S:SAVE      E:END

?
```

Fig. 5-1. Creating spreadsheet, steps 1-4.

```
** TI-99/4A   CALC ** 97% **

         A                   B

A       AUTO GAS>MILEAGE       <
B  ------------ ------------
C
D  START MILES:
E    END MILES:
F  GALLONS GAS:
G
H              MPG +EB-DB/FB
I
J
K
L
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 5-2. Creating spreadsheet, steps 5-9.

```
** TI-99/4A   CALC ** 96% **

         A                   B

A       AUTO GAS MILEAGE
B  ------------ ------------
C
D  START MILES:        22170.00
E    END MILES:        24270.00
F  GALLONS GAS: >          70.00<
G
H              MPG          30.00
I
J
K
L
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 5-3. First calculation, steps 10-14.

CHAPTER SIX
Sample *TI CALC* Spreadsheets

In some respects the BASIC language is similar to a program like *TI CALC*. Both permit you to use the computer to perform many different functions. They are each "software tools" that enable you to "instruct" the computer to do many different operations that together produce a desired result.

For example, you can write a BASIC program that will convert degrees in Fahrenheit to degrees in Celsius. To do this in BASIC you must become familiar with some of the programming considerations involved in using that particular language. However, you can also write a *TI CALC* spreadsheet that will make this same temperature conversion for you. In BASIC you write a program; in *TI CALC* you write a spreadsheet.

To show you ways to use *TI CALC* and to help you become familiar with setting up spreadsheets, there are five sample spreadsheets in this chapter. There is a section on each that includes instructions for entering the spreadsheet and then a discussion on how to use the spreadsheet. All of the sample spreadsheets (plus "Auto Gas Mileage", see Chapter 5 for a complete walkthru) are also included on the cassette tape so you can simply load them directly into *TI CALC*.

If you purchased the book only, each of the sample spreadsheets can be entered by inputting the cell entries exactly as shown in Figs. 6-5, 6-11, 6-16, 6-21, and 6-26.

The five sample spreadsheets included are:

1. Temperature conversion
2. Piggy bank counter
3. Income planning
4. Loan analysis
5. Savings plan

Temperature Conversion

FUNCTION: Performs the temperature conversion either way, Fahrenheit to Celsius, or Celsius to Fahrenheit. You input the temperature in one scale and *TI CALC* will display the equivalent temperature in the other scale. The initial screen is shown in Fig. 6-1. Fig. 6-2 shows the rest of the initial screen.

```
** TI-99/4A   CALC ** 94% **

         A                  B

A>  TEMPERATURE<CONVERSION
B  ------------ ------------
C
D     FAHRENHEIT TO CELSIUS
E
F    ENTER D.F.:
G    RESULT D.C.  +FB-DC*EC/FC
H
I
J     CELSIUS TO FAHRENHEIT
K
L    ENTER D.C.:
M    RESULT D.F.  +LB*FC/EC+DC

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-1. Temperature conversion initial screen.

```
**  TI-99/4A   CALC  **  94%  **

          C                   D

A>  ADDITIONAL  <
B     NUMBERS
C
D             32.00
E              5.00
F              9.00
G
H
I
J
K
L
M

G:GOTO       C:CALC
N:NUMBER     T:TEXT       F:FORMULA
L:LOAD       S:SAVE       E:END

?
```

Fig. 6-2. Temperature conversion second screen.

INPUT: There are only two input or "what if" variables. One for each temperature scale. To input a Fahrenheit temperature, position the active cell to cell FB and enter the value. To input a Celsius temperature, position the active cell to cell LB and enter the value.

OUTPUT: *TI CALC* will also output two variable cells, one for each temperature scale. The two output cells are cell GB and cell MB, where these two cells contain the converted temperature values.

OPERATION: To use this spreadsheet you must first position the active cell to one of the input or "what if" variables and enter the temperature value you would like converted. Input the value with the **NUMBER** command. After the temperature is entered, type in the **CALC** command and *TI CALC* will perform the conversion and display the results.

NOTES: If you do not enter a value in one of the input cells, *TI CALC* will assume the value of zero and do the conversion based on zero. So even though you do not enter a value, you may see a result displayed that you did not expect.

EXAMPLES: As a test, this example has you enter 32 for degrees in Fahrenheit and on the bottom half of the screen you enter 0 for degrees Celsius. What *TI CALC* should calculate is 0 degrees Celsius

for 32 degrees Fahrenheit and 32 degrees Fahrenheit for 0 degrees Celsius. After you enter these input values the screen should look like Fig. 6-3.

```
** TI-99/4A   CALC ** 94% **

         A                  B

A   TEMPERATURE CONVERSION
B  ------------ ------------
C
D    FAHRENHEIT TO CELSIUS
E
F   ENTER D.F.:          32.00
G   RESULT D.C.  +FB-DC*EC/FC
H
I
J    CELSIUS TO FAHRENHEIT
K
L   ENTER D.C.:>          0.00<
M   RESULT D.F.  +LB*FC/EC+DC

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-3. Temperature conversion, input.

After you use the CALC command, you can see that the conversion is accurate. You may want to enter a temperature that is not so easily converted. For example, you might want to enter the value of 92 degrees Fahrenheit in cell FB and perform a CALC command. See Fig. 6-4.

```
** TI-99/4A   CALC ** 94% **

         A                   B

A   TEMPERATURE CONVERSION
B  ------------ ------------
C
D    FAHRENHEIT TO CELSIUS
E
F   ENTER D.F.: >          92.00<
G   RESULT D.C.            33.33
H
I
J    CELSIUS TO FAHRENHEIT
K
L   ENTER D.C.:             0.00
M   RESULT D.F.            32.00

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-4. Temperature conversion, recalculated.

As it turns out, 92 degrees Fahrenheit is the same as 33.33 degrees Celsius.

```
AA=" TEMPERATURE"          T
BA="-----------"           T
DA="  FAHRENHEIT"          T
FA=" ENTER D.F.:"          T
GA=" RESULT D.C."          T
JA="  CELSIUS TO"          T
LA=" ENTER D.C.:"          T
MA=" RESULT D.F."          T
AB="CONVERSION"            T
BB="-----------"           T
DB="TO CELSIUS"            T
GB=+FB-DC*EC/FC            F
JB="FAHRENHEIT"            T
MB=+LB*FC/EC+DC            F
AC=" ADDITIONAL"           T
BC="  NUMBERS"            T
DC=32                      N
EC=5                       N
FC=9                       N
```

Fig. 6-5 Temperature conversion, cell entries.

Piggy Bank Counter

FUNCTION: Permits you to add various and large amounts of change so that you can arrive at a total without losing track of your count.

```
** TI-99/4A   CALC ** 92% **

         A                   B

A>   PIGGY BANK<COUNTER
B  ------------ ------------
C
D     # PENNIES:
E     # NICKELS:
F       # DIMES:
G   # QUARTERS:
H # HALF-DLRS:
I     # DOLLARS:
J  ------------ ------------
K
L       TOTAL     $ +ID/IC
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-6. Piggy bank counter initial screen.

```
** TI-99/4A   CALC ** 92% **

         C                   D

A>  ADDITIONAL < ADDITIONAL
B     NUMBERS       FORMULAS
C
D
E             5.00  +EB*EC+DB
F            10.00  +FB*FC+ED
G            25.00  +GB*GC+FD
H            50.00  +HB*HC+GD
I           100.00  +IB*IC+HD
J
K
L
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-7. Piggy bank counter second screen.

INPUT: There are six input variables or "what if" cells on the screen—one for each type of coin. Just move to the correct cell in column B and enter the number of pennies, nickels, dimes, quarters, half-dollars, and dollars.

OUTPUT: The total amount of money in dollars and cents will be displayed at cell LB.

OPERATION: Simply put the number of coins in the "what if" areas (cells DB-IB) using the **NUMBER** command. If you don't have one type of coin, you don't have to enter anything in that cell because it will default to zero. Press the **CALC** command and *TI CALC* will total the amount of money and display results. Column C holds conversion values and column D keeps a cumulative total during the CALC routine.

EXAMPLES: As a test of this example, enter the number of coins needed to total exactly one dollar for each type of coin. For example, 100 pennies, 20 nickels, 10 dimes, 4 quarters, 2 half-dollars, and one dollar. *TI CALC* should total six dollars. After you enter the input values, the screen will appear as you see it in Fig 6-8.

```
**  TI-99/4A   CALC  **  91%  **

          A                    B

A     PIGGY BANK COUNTER
B  ------------  ------------
C
D     #  PENNIES:          100.00
E     #  NICKELS:           20.00
F        #  DIMES:          10.00
G   #  QUARTERS:             4.00
H  #  HALF-DLRS:             2.00
I     #  DOLLARS: >          1.00<
J  ------------  ------------
K
L       TOTAL     $  +ID/IC
M

G: GOTO      C: CALC
N: NUMBER    T: TEXT      F: FORMULA
L: LOAD      S: SAVE      E: END

?
```

Fig. 6-8. Piggy bank counter, input.

After pressing the CALC command, the screen will appear as in Fig. 6-9.

```
**  TI-99/4A   CALC  **  91%  **

          A                   B

A     PIGGY BANK COUNTER
B  ------------  ------------
C
D     #  PENNIES:          100.00
E     #  NICKELS:           20.00
F        #  DIMES:          10.00
G   #  QUARTERS:             4.00
H  #  HALF-DLRS:             2.00
I     #  DOLLARS: >          1.00<
J  ------------  ------------
K
L       TOTAL     $           6.00
M

G: GOTO      C: CALC
N: NUMBER    T: TEXT     F: FORMULA
L: LOAD      S: SAVE     E: END

?
```

Fig. 6-9. Piggy bank counter, calculated.

Now try doing the same thing using two dollars for each type of coin. Enter 200 pennies, 40 nickels, and so on. Then press the CALC command and the screen will look as it does in Fig. 6-10.

```
** TI-99/4A  CALC ** 91% **

        A                  B

A    PIGGY BANK COUNTER
B  ------------ ------------
C
D     # PENNIES:          200.00
E     # NICKELS:           40.00
F       # DIMES:           20.00
G   # QUARTERS:             8.00
H # HALF-DLRS:              4.00
I     # DOLLARS: >           2.00<
J  ------------ ------------
K
L      TOTAL    $            12.00
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-10. Piggy bank counter, recalculated.

```
AA="  PIGGY BANK"          T
BA="------------"          T
DA="  # PENNIES:"          T
EA="  # NICKELS:"          T
FA="    # DIMES:"          T
GA=" # QUARTERS:"          T
HA="# HALF-DLRS:"          T
IA="  # DOLLARS:"          T
JA="------------"          T
LA="   TOTAL   $"          T
AB="COUNTER"               T
BB="------------"          T
JB="------------"          T
LB=+ID/IC                  F
AC=" ADDITIONAL"           T
BC="  NUMBERS"             T
EC=5                       N
FC=1Ø                      N
GC=25                      N
HC=5Ø                      N
IC=1ØØ                     N
AD=" ADDITIONAL"           T
BD="  FORMULAS"           T
ED=+EB*EC+BD               F
FD=+FB*FC+ED               F
GD=+GB*GC+FD               F
HD=+HB*HC+GD               F
ID=+IB*IC+HD               F
```

Fig. 6-11. Piggy bank counter, cell entries.

Income Planning

FUNCTION: This spreadsheet will allow you to see what your gross pay and net pay would be at various different hourly rates and with differing amounts of hours worked per week. In addition, your gross and net pay are also calculated for a month and for a year based on the given hours per week and rate per hour. See Fig. 6-12 for initial screen.

```
** TI-99/4A   CALC ** 93% **

         A                   B

A>         INCOME<PLANNING
B  ------------  ------------
C
D   HOURS/WEEK:
E    RATE/HOUR:
F
G  PAY --   WEEK  +DB*EB
H          MONTH  +GB*DC
I           YEAR  +GB*EC
J
K  NET --   WEEK  +GB*HC
L          MONTH  +HB*HC
M           YEAR  +IB*HC

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-12. Income planning initial screen.

```
** TI-99/4A   CALC ** 93% **

         C                   D

A> ADDITIONAL <
B   NUMBERS
C
D           4.00
E          52.00
F
G    NET RATE
H             .80
I
J
K
L
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-13. Income planning second screen.

INPUT: There are two input cells. Cell DB will hold the hours per week worked and cell EB will hold the rate per hour.

OUTPUT: Based on only two input cells this spreadsheet will calculate a total of six other cells. This many calculated cells begins to show you some of the power of *TI CALC* and of spreadsheets in general.

OPERATION: Simply position the active cell to the two input cells and enter your desired hours per week and rate per hour. Enter these values with the **NUMBER** command. When you have entered each, then perform a **CALC** command and the results will be displayed on the screen.

NOTES: In the previous sample spreadsheet you could omit one of the input cells and still get the correct results. However, with this spreadsheet each of the two input cells should be given values for any meaningful results to be calculated. The tax rate is assumed to be 20 percent. If this does not give correct information you may try adjusting this number to what is best for you. The tax rate can be found at cell HC (20% tax rate means .80 net rate).

EXAMPLES: For starters, if you enter 40.00 for hours per week and 8.00 for rate per hour and then perform the **CALC** command the screen would appear as it does in Fig. 6-14.

```
** TI-99/4A  CALC ** 92% **

        A                  B

A          INCOME PLANNING
B ------------ ------------
C
D  HOURS/WEEK:          40.00
E   RATE/HOUR: >         8.00<
F
G PAY --  WEEK          320.00
H        MONTH         1280.00
I         YEAR        16640.00
J
K NET --  WEEK          256.00
L        MONTH         1024.00
M         YEAR        13312.00

G:GOTO     C:CALC
N:NUMBER   T:TEXT     F:FORMULA
L:LOAD     S:SAVE     E:END

?
```

Fig. 6-14. Income planning, calculated.

Now if you were expecting a nice .75 cent raise you might want to change the rate per hour to 8.75 and recalculate the spreadsheet. At 8.75 for the rate per hour the results are as you see in Fig. 6-15.

```
** TI-99/4A   CALC ** 92% **

         A                    B

A          INCOME PLANNING
B  ------------ ------------
C
D   HOURS/WEEK:            40.00
E    RATE/HOUR: >           8.75<
F
G  PAY --   WEEK           350.00
H          MONTH          1400.00
I           YEAR         18200.00
J
K  NET --   WEEK           280.00
L          MONTH          1120.00
M           YEAR         14560.00

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-15. Income planning, recalculated.

```
AA="       INCOME"          T
BA="------------"           T
DA=" HOURS/WEEK:"           T
EA="  RATE/HOUR:"           T
GA="PAY --  WEEK"           T
HA="        MONTH"          T
IA="         YEAR"          T
KA="NET --  WEEK"           T
LA="        MONTH"          T
MA="         YEAR"          T
AB="PLANNING"               T
BB="------------"           T
GB=+DB*EB                   F
HB=+GB*DC                   F
IB=+GB*EC                   F
KB=+GB*HC                   F
LB=+HB*HC                   F
MB=+IB*HC                   F
AC=" ADDITIONAL"            T
BC="  NUMBERS"              T
DC=4                        N
EC=52                       N
GC="  NET RATE"            T
HC=.8                       N
```

Fig. 6-16. Income planning, cell entries.

Loan Analysis

FUNCTION: Loan Analysis helps you evaluate a loan by permitting you to vary such amounts as the principal, the interest rate, and the number of months to repay. Based on these "what if" variables, the spreadsheet will calculate three other variables (or cells)—the monthly payment, the total amount repaid, and the total interest amount charged.

```
** TI-99/4A   CALC ** 93% **

         A                   B

A>           LOAN<ANALYSIS
B  ------------ ------------
C
D  LOAN AMOUNT:
E      INT RATE:
F  #  OF MONTHS:
G
H          CHANGE ABOVE
I         RESULTS BELOW
J
K  MNTHLY PYMNT +DB*EC/HC
L    TOTAL LOAN +KB*FB
M  TOTAL INTRST +LB-DB

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-17. Loan analysis initial screen.

```
** TI-99/4A   CALC ** 93% **

         C                   D

A>  ADDITIONAL <
B     FORMULAS
C             1.00
D          1200.00
E   +EB/DC
F   +CC+EC^FB
G   +CC/FC
H   +CC-GC
I
J
K
L
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT      F:FORMULA
L:LOAD      S:SAVE      E:END

?
```

Fig. 6-18. Loan analysis second screen.

INPUT: The three "what if" variables are located on the top half of the spreadsheet in column B and they are labeled in column A: "LOAN AMOUNT", "INT RATE", and "# OF MONTHS".

OUTPUT: The three output cells are located on the bottom half of the spreadsheet in column B and are labeled in column A: "MNTHLY PYMNT", "TOTAL LOAN", and "TOTAL INTRST".

OPERATION: The first thing to do is to set the three "what if" variables to initial values. This is done by positioning the active cell first to cell DB and entering the loan amount. Then advance the active cell to the other two input cells and enter their initial values. Once they are set, enter the **CALC** command to display the results. After that you can change any or all of the three "what if" variables to find the best loan for you.

In column C there are some special numbers and formulas that are used by the spreadsheet for its various calculations.

EXAMPLES: To begin, enter 5000 for the loan amount, 12 for the interest rate, and 36 for the number of repayment months. After you've entered those in cells DB, EB, and FB, respectively, you can enter the **CALC** command. When you've done that, the screen will appear as it does in Fig. 6-19.

```
** TI-99/4A  CALC ** 91% **

         A                  B

A            LOAN ANALYSIS
B ------------ ------------
C
D LOAN AMOUNT:          5000.00
E     INT RATE:            12.00
F # OF MONTHS: >           36.00<
G
H          CHANGE ABOVE
I         RESULTS BELOW
J
K MNTHLY PYMNT           166.07
L   TOTAL LOAN          5978.57
M TOTAL INTRST            978.57

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-19. Loan analysis, calculated.

You may then want to see what the figures would look like if you were to repay the loan in 48 months instead of 36. To see the results of this, simply position the active cell to cell FB and use the NUMBER command to change the entry to 48. Then do another CALC command. The screen will appear as it does in Fig. 6-20.

```
** TI-99/4A   CALC ** 91% **

         A                   B

A            LOAN ANALYSIS
B  ------------  ------------
C
D  LOAN AMOUNT:          5000.00
E     INT RATE:            12.00
F  # OF MONTHS: >          48.00<
G
H          CHANGE ABOVE
I         RESULTS BELOW
J
K  MNTHLY PYMNT           131.66
L    TOTAL LOAN          6320.12
M  TOTAL INTRST          1320.12

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-20. Loan analysis, recalculated.

```
AA="          LOAN"          T
BA="------------"            T
DA="LOAN AMOUNT:"            T
EA="   INT RATE:"            T
FA="# OF MONTHS:"            T
HA="       CHANGE"           T
IA="      RESULTS"           T
KA="MNTHLY PYMNT"            T
LA="  TOTAL LOAN"            T
MA="TOTAL INTRST"            T
AB="ANALYSIS"                T
BB="------------"            T
HB="ABOVE"                   T
IB="BELOW"                   T
KB=+DB*EC/HC                 F
LB=+KB*FB                    F
MB=+LB-DB                    F
AC=" ADDITIONAL"             T
BC="  FORMULAS"              T
CC=1                         N
DC=12ØØ                      N
EC=+EB/DC                    F
FC=+CC+EC^FB                 F
GC=+CC/FC                    F
HC=+CC-GC                    F
```

Fig. 6-21. Loan analysis, cell entries.

Savings Plan

FUNCTION: This spreadsheet will show the amount of money per year you could save based on a monthly amount saved and a specified interest rate. See Figs. 6-22 and 6-23 for the initial screens.

```
** TI-99/4A   CALC ** 92% **

         A                   B

A>       SAVINGS<PLAN
B  ------------ ------------
C
D  WKLY AMOUNT:
E      INT RATE:
F
G
H     SAVINGS
I  ------------
J     SIX MONTHS  +GC^IC-BC*HC
K       ONE YEAR  +GC^JC-BC*HC
L      TWO YEARS  +GC^KC-BC*HC
M     FIVE YEARS  +GC^LC-BC*HC

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-22. Savings plan initial screen.

```
** TI-99/4A   CALC ** 92% **

         C                   D

A> EXTRA INFO <
B           1.00
C           7.00
D         100.00
E         365.00
F  +EB/DC/EC
G  +FC+BC
H  +DB/CC/FC
I         182.50
J         365.00
K         730.00
L        1825.00
M

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-23. Savings plan second screen.

INPUT: There are two input cells for this spreadsheet, the amount per week to save and the interest rate. The amount per week is at cell DB and the interest rate is at cell EB.

OUTPUT: There are four output cells that are calculated for four different periods of saving. Cell JB is for after 6 months, KB is one year, LB is two years, and MB is five years.

OPERATION: To use this spreadsheet position the active cell to the input cells for the amount per week saved and the interest rate. Enter these two values with the **NUMBER** command. Once these values are entered, then perform the **CALC** command and the results will be displayed.

NOTES: The input cells are on the top half of the spreadsheet and the output cells are on the bottom half.

EXAMPLES: Let's start off by assuming that you can save $10.00 per week. Also, let's use an interest rate of 9.00 percent. After you entered these values and performed the **CALC** command the screen would be displayed as you see it in Fig. 6-24.

```
**  TI-99/4A   CALC  **  90%  **

          A                   B

A         SAVINGS PLAN
B  ------------  -------------
C
D  WKLY AMOUNT:             10.00
E      INT RATE: >           9.00<
F
G
H     SAVINGS
I  ------------
J     SIX MONTHS            266.63
K       ONE YEAR            545.54
L      TWO YEARS           1142.45
M     FIVE YEARS           3292.09

G: GOTO      C: CALC
N: NUMBER    T: TEXT     F: FORMULA
L: LOAD      S: SAVE     E: END

?
```

Fig. 6-24. Savings plan, calculated.

If you are not too pleased with the amount saved at $10.00 per week

you might try to save $15.00 per week. The results of this change would appear as they do in Fig. 6-25.

```
** TI-99/4A   CALC ** 90% **

          A                   B

A         SAVINGS PLAN
B  ------------  ------------
C
D  WKLY AMOUNT:            15.00
E      INT RATE:>           9.00<
F
G
H    SAVINGS
I  ------------
J     SIX MONTHS           399.95
K       ONE YEAR           818.31
L      TWO YEARS          1713.67
M     FIVE YEARS          4938.13

G:GOTO      C:CALC
N:NUMBER    T:TEXT     F:FORMULA
L:LOAD      S:SAVE     E:END

?
```

Fig. 6-25. Savings plan, recalculated.

At this point you could leave the amount per week at $15.00 and change the interest rate, perhaps to compare banks.

```
AA="      SAVINGS"          T
BA="------------"           T
DA="WKLY AMOUNT:"           T
EA="   INT RATE:"           T
HA="  SAVINGS"             T
IA="------------"           T
JA="  SIX MONTHS"          T
KA="    ONE YEAR"          T
LA="   TWO YEARS"          T
MA="  FIVE YEARS"          T
AB="PLAN"                   T
BB="------------"           T
JB=+GC^IC-BC*HC             F
KB=+GC^JC-BC*HC             F
LB=+GC^KC-BC*HC             F
MB=+GC^LC-BC*HC             F
AC=" EXTRA INFO"            T
BC=1                        N
CC=7                        N
DC=100                      N
EC=365                      N
FC=+EB/DC/EC                F
GC=+FC+BC                   F
HC=+DB/CC/FC                F
IC=182.5                    N
JC=365                      N
KC=730                      N
LC=1825                     N
```

Fig. 6-26. Savings plan, cell entries.

CHAPTER SEVEN
THE *TI CALC* PROGRAM

Where Do You Start?

Where do you start to write a program? Well you start at the beginning of course. Fine, but where is that? The beginning is very simply an "idea". That may sound too simple or perhaps a little corny, but it is true. The original idea for *TI CALC* was that there were no spreadsheet programs written for the TI-99/4A home computer (with the exception of MicroSoft's MultiPlan, but that needs the expansion box, 32K of memory, and a disk drive, not exactly a typical home setup). My thoughts were that spreadsheet programs were so popular and so useful that there was a definite need for one on the TI-99/4A home computer. However, this spreadsheet program would have to run on the most basic of computer configurations. The most common TI configuration is the TI-99/4A computer, a television, and a cassette recorder. Because of these constraints, it would have to be a subset of the other larger spreadsheet programs but still provide the standard spreadsheet functions so it would be useful.

Now that I'd thought of the idea, what was I to do next? Good question, because at this point nothing was down on paper and nothing was in the computer. The only thing I had was the idea. Next came the most difficult part, designing the spreadsheet pro-

gram. At this design stage I determined the limits and capabilities of the program—how many cells should it support, what symbols should be used as row and column identifiers, how would the program be stored internally, how many columns should be displayed at one time on the screen, what arithmetic functions should it support, how many calculations should be allowed to be embedded within each other, and so on.

The point is that everything needs to be determined—first at a general level, then more specifically, and then finally at the level where the programmer begins to write the BASIC code.

Creating a program is completely different from taking an existing program and making changes or enhancements to it. For example, when I was in college there was a football program on the computer system. Well, the program was fun, but I thought it would be better if it could provide more colorful descriptions of each play. So I made changes and additions to the football program. I included a narrative of each play and, at the end of the game, the complete game statistics were displayed. Making these changes was rather straightforward because the main program was already written and I did not have to determine all of the things that the original developers of the program did. Of course, just because a program is written and runs correctly does not mean it is complete. As a matter of fact, programs are constantly being changed and enhanced. It is at this stage in the design of a program that an effort should be made to make future changes as easy as possible. So where do you start? Once again, you start with an idea. Think about what types of things you'd want the program to be able to do in the future.

Design of *TI CALC*

As noted earlier, *TI CALC* is a subset of some of the larger spreadsheet programs. Because of the 16K memory limitation in the basic TI-99/4A configuration, compromises and tradeoffs were made to provide a spreadsheet program that was still functional. The biggest compromise was in the total number of cells. The *TI CALC* matrix is 26 rows by 26 columns, a total of 676 cells. There are many spreadsheet programs that allow several thousand different cell locations, but these spreadsheets run on machines that usually have at least 128K of memory and at least one disk drive. However, if *TI CALC* allocated much more memory for the data there would be no room for the program. As it turns out, a 26 by 26 matrix works out rather

nicely. This decision allowed me to use letters of the alphabet as both the row and column identifiers.

In the design of *TI CALC* there were three main design considerations. The first was the matrix size and how it would be displayed on the screen. For example, how many rows and columns should be displayed at one time. Once the matrix size was determined to be 26 by 26, a logical display for the number of rows was 13 rows, or half the number of rows in the matrix. Determining the number of columns to be displayed was not as easy. However, once it was decided to keep a certain number of columns displayed at all times and not be able to change the number of columns displayed, there was only one choice. Two columns displayed at all times was the only way to go. Two columns allows up to 12 positions per cell. Three columns per cell might have cut us a little short with only 8 positions, and one column per display was far too restrictive a view into the spreadsheet.

The second main design consideration was how the matrix would be stored internally while in memory. The strategy I adopted was to use two 26 by 26 string arrays. The first matrix is used to hold all text entries and all formulas. The second matrix is used to hold the numbers and the results of the formulas once a calculation was performed. The reason this second array is needed is to allow embedding of calculations (a calculation which points to another calculation). Implemented in this way, *TI CALC* makes no distinction between an embedded calculation or a regular calculation. Thus, there are no further considerations that *TI CALC* need be "concerned" with when a calculation is embedded. This provides us with two very nice features. First, you may embed any number of calculations, and second, embedding a calculation does not slow *TI CALC* down.

The third main design consideration was how the user of the program would enter data and commands into the spreadsheet. Spreadsheet programs are all noted for their capability to move the cursor, or active cell, anywhere on the screen. I decided that this capability was needed to make *TI CALC* easier to use. Also, a command menu was added to make the commands easier to use. The use of menus is becoming more and more popular. The reason? You do not need to memorize any commands because they are all on the screen. *TI CALC* has eight commands and all eight are on the screen at all times. The combination of being able to move the active cell and the

presence of the command menu provides an environment that is typical of all spreadsheets.

Possible Advanced Features

TI CALC is a fully functional spreadsheet program and, therefore, you can use *TI CALC* to develop your own spreadsheets. It can also be used as a type of educational program, because once you learn how to use *TI CALC* it will be easier for you to learn one of the larger spreadsheet programs.

Perhaps you have a spreadsheet program available at school or at work. Once you understand the concept of a spreadsheet and how to make it do what you want, this knowledge can be used when you work with other spreadsheet programs. *TI CALC* provides the fundamental spreadsheet functions, but there are many features that other spreadsheet programs have and *TI CALC* does not. Some of the more common advanced features found on other spreadsheet programs are:

1. 256 rows by 64 columns (16384 cells)
2. capability to use numbers directly in the calculation
3. a function to sum a column of numbers
4. a function to copy a formula to other cells
5. capability to link spreadsheets together
6. capability to sort a column
7. capability to add or delete a row or column
8. capability to hold a row or column on the screen
9. capability to protect a cell with a formula in it
10. speed, with more memory and faster central processing

Many of the above listed features are very nice, but I think the last is the most important. Unfortunately, TI BASIC is slow, and because it is slow the spreadsheet program, at times, is also slow. Even though every effort has been made to make the code as efficient as possible, sometimes *TI CALC* is not as fast as I would like. Spreadsheet programs are usually written in machine language which makes for high speed operation. This lack of speed was one of the compromises that was made to provide a spreadsheet program on the TI-99/4A computer.

The other advanced features were noted so that you would have a better idea of the capabilities of other spreadsheets. If you have a spreadsheet at work it will probably have many of these featues.

Overview of the Program

As noted earlier, the design stage is when certain efforts should be made to make future changes as easy as possible. The main feature of a program that makes changes easier is to program in modules. Each module (logical group of BASIC lines) performs a specific function. For example, there is a module that performs the load function of the **LOAD** command. The sole purpose of this module is to only load spreadsheets into *TI CALC*.

There are ten main modules in *TI CALC* and they are as follows:

1. Spreadsheet print routine
2. Command getting routine
3. GOTO routine
4. EXIT routine
5. CALC routine
6. LOAD routine
7. SAVE routine
8. Cursor movement routine
9. Cell editing routine
10. Spreadsheet format print routine

For a detailed discussion on the program please see the end of this chapter.

Program Variable Usage

The variables that are used in *TI CALC* are listed below:

1. A$(26,26) — Text and formulas
2. B$(26,26) — All numbers
3. X,Y — Upperleftmost row, column
4. D,E — Row, column displacement from cell x, y
5. T$ — Text or formula of current cell
6. V$ — Value of current cell, for numbers
7. R,C — Row and column loop variables
8. G,H — Greatest row and column used
9. K,S — Call key variables
10. M,N,M$ — Message printing variables
11. Z — Accumulator for the CALC routine
12. O — Operation type
13. I,J — Row, column in formulas
14. T — Temporary variable in calculations
15. L — Loop variable

16. F — Flag in editing routine
17. Q — Amount of free memory
18. P — Temporary variable in editing routine
19. L$ — Temporary variable for formatting and printing routines

The *TI CALC* Program

```
100 DIM A$(26,26),B$(26,26)
110 X=1
120 Y=1
130 Q=5000
140 CALL CLEAR
150 IF Y<26 THEN 180
160 E=1
170 Y=25
180 IF X<15 THEN 210
190 D=X-14
200 X=14
210 PRINT "** TI-99/4A  CALC ** ";STR$(INT(Q/50));"% *
    *";::TAB(8);CHR$(Y+64);TAB(22);CHR$(Y+65);::
220 FOR R=X TO X+12
230 T$=A$(R,Y)
240 V$=B$(R,Y)
250 GOSUB 2220
260 M$=L$
270 T$=A$(R,Y+1)
280 V$=B$(R,Y+1)
290 GOSUB 2220
300 PRINT CHR$(R+64);" ";M$;TAB(16);L$
310 NEXT R
320 PRINT :"G:GOTO     C:CALC":"N:NUMBER  T:TEXT    F:FO
    RMULA":"L:LOAD     S:SAVE    E:END":::
330 K=0
340 GOTO 1520
350 IF X+D<G THEN 370
360 G=X+D
370 IF Y+E<H THEN 390
380 H=Y+E
390 T$=A$(X+D,Y+E)
400 V$=B$(X+D,Y+E)
410 CALL HCHAR(24,3,32,28)
420 CALL HCHAR(24,3,63)
430 CALL KEY(3,K,S)
440 IF S=0 THEN 430
450 CALL HCHAR(24,4,K)
```

```
460 IF (K>7)*(K<12)THEN 1520
470 IF (K=76)+(K=83)+(K=69)THEN 700
480 IF K=71 THEN 510
490 IF K=67 THEN 810
500 IF (K=78)+(K=84)+(K=70)THEN 1720 ELSE 420
510 M=24
520 N=7
530 M$="ROW?"
540 GOSUB 1680
550 CALL KEY(3,K,S)
560 IF (K<65)+(K>90)THEN 550
570 CALL HCHAR(24,12,K)
580 X=K-64
590 M=24
600 N=16
610 M$="COLUMN?"
620 GOSUB 1680
630 CALL KEY(3,K,S)
640 IF (K<65)+(K>90)THEN 630
650 CALL HCHAR(24,24,K)
660 Y=K-64
670 D=0
680 E=0
690 GOTO 140
700 F=K
710 M=24
720 N=7
730 M$="ARE YOU SURE (Y/N)?"
740 GOSUB 1680
750 CALL KEY(3,K,S)
760 IF K=89 THEN 780
770 IF K=78 THEN 350 ELSE 750
780 CALL CLEAR
790 IF F=69 THEN 2370
800 IF F=76 THEN 1220 ELSE 1400
810 FOR R=1 TO G
820 FOR C=1 TO H
830 IF SEG$(A$(R,C),1,1)="+" THEN 870
840 NEXT C
850 NEXT R
860 GOTO 140
870 Q=Q+LEN(B$(R,C))-((LEN(B$(R,C))>0)*4)
880 Z=0
890 T$=A$(R,C)
900 FOR L=1 TO LEN(T$)STEP 3
910 T=0
920 O=ASC(SEG$(T$,L,1))
930 I=ASC(SEG$(T$,L+1,1))-64
940 J=ASC(SEG$(T$,L+2,1))-64
950 IF (I<1)+(I>26)+(J<1)+(J>26)+((O<>42)*(O<>43)*(O<>
    45)*(O<>47)*(O<>94))THEN 1150
960 IF B$(I,J)<>"" THEN 970 ELSE 980
970 T=VAL(B$(I,J))
980 IF O=94 THEN 1000
990 ON O-41 GOTO 1020,1040,1150,1060,1150,1080
```

```
1000 Z=Z^T
1010 GOTO 1100
1020 Z=Z*T
1030 GOTO 1100
1040 Z=Z+T
1050 GOTO 1100
1060 Z=Z-T
1070 GOTO 1100
1080 IF T=0 THEN 1150
1090 Z=Z/T
1100 NEXT L
1110 IF POS(STR$(Z),"*",1)>1 THEN 1150
1120 B$(R,C)=STR$(Z)
1130 Q=Q-LEN(B$(R,C))+((LEN(B$(R,C))>0)*4)
1140 GOTO 840
1150 Q=Q-LEN(B$(R,C))+((LEN(B$(R,C))>0)*4)
1160 M=24
1170 N=3
1180 M$=" ERROR AT "&CHR$(R+64)&CHR$(C+64)&" - PRESS A
     NY KEY"
1190 GOSUB 1680
1200 CALL KEY(3,K,S)
1210 IF S=0 THEN 1200 ELSE 140
1220 PRINT "CALC LOAD"
1230 FOR R=1 TO G
1240 FOR C=1 TO H
1250 A$(R,C)=""
1260 B$(R,C)=""
1270 NEXT C
1280 NEXT R
1290 OPEN #1:"CS1",INTERNAL,INPUT ,FIXED 192
1300 INPUT #1:G,H,Q
1310 FOR C=1 TO H
1320 INPUT #1:A$(1,C),A$(2,C),A$(3,C),A$(4,C),A$(5,C),
     A$(6,C),A$(7,C),A$(8,C),A$(9,C),A$(10,C),A$(11,C)
     ,A$(12,C),A$(13,C)
1330 INPUT #1:B$(1,C),B$(2,C),B$(3,C),B$(4,C),B$(5,C),
     B$(6,C),B$(7,C),B$(8,C),B$(9,C),B$(10,C),B$(11,C)
     ,B$(12,C),B$(13,C)
1340 IF G<14 THEN 1370
1350 INPUT #1:A$(14,C),A$(15,C),A$(16,C),A$(17,C),A$(1
     8,C),A$(19,C),A$(20,C),A$(21,C),A$(22,C),A$(23,C)
     ,A$(24,C),A$(25,C),A$(26,C)
1360 INPUT #1:B$(14,C),B$(15,C),B$(16,C),B$(17,C),B$(1
     8,C),B$(19,C),B$(20,C),B$(21,C),B$(22,C),B$(23,C)
     ,B$(24,C),B$(25,C),B$(26,C)
1370 NEXT C
1380 CLOSE #1
1390 GOTO 140
1400 PRINT "CALC SAVE"
1410 OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED 192
1420 PRINT #1:G,H,Q
1430 FOR C=1 TO H
1440 PRINT #1:A$(1,C),A$(2,C),A$(3,C),A$(4,C),A$(5,C),
```

```
     A$(6,C),A$(7,C),A$(8,C),A$(9,C),A$(10,C),A$(11,C)
     ,A$(12,C),A$(13,C)
1450 PRINT #1:B$(1,C),B$(2,C),B$(3,C),B$(4,C),B$(5,C),
     B$(6,C),B$(7,C),B$(8,C),B$(9,C),B$(10,C),B$(11,C)
     ,B$(12,C),B$(13,C)
1460 IF G<14 THEN 1490
1470 PRINT #1:A$(14,C),A$(15,C),A$(16,C),A$(17,C),A$(1
     8,C),A$(19,C),A$(20,C),A$(21,C),A$(22,C),A$(23,C)
     ,A$(24,C),A$(25,C),A$(26,C)
1480 PRINT #1:B$(14,C),B$(15,C),B$(16,C),B$(17,C),B$(1
     8,C),B$(19,C),B$(20,C),B$(21,C),B$(22,C),B$(23,C)
     ,B$(24,C),B$(25,C),B$(26,C)
1490 NEXT C
1500 CLOSE #1
1510 GOTO 140
1520 CALL HCHAR(D+5,E*13+4,32)
1530 CALL HCHAR(D+5,E*13+17,32)
1540 IF ((K=8)*(E=1))+((K=9)*(E=0))THEN 1580
1550 IF ((K=8)*(E=0)*(Y>1))+((K=9)*(E=1)*(Y<25))THEN 1
     640
1560 IF ((K=10)*(D<12))+((K=11)*(D>0))THEN 1600
1570 IF ((K=10)*(D=12)*(X<14))+((K=11)*(D=0)*(X>1))THE
     N 1660 ELSE 1610
1580 E=1-E
1590 GOTO 1610
1600 D=D+21-K*2
1610 CALL HCHAR(D+5,E*13+4,62)
1620 CALL HCHAR(D+5,E*13+17,60)
1630 GOTO 350
1640 Y=Y+E*2-1
1650 GOTO 140
1660 X=X+21-K*2
1670 GOTO 140
1680 FOR L=1 TO LEN(M$)
1690 CALL HCHAR(M,N+L-1,ASC(SEG$(M$,L,1)))
1700 NEXT L
1710 RETURN
1720 F=K
1730 Q=Q+LEN(V$)+LEN(T$)-((LEN(V$)>0)*4)-((LEN(T$)>0)*
     4)
1740 CALL HCHAR(24,7,62)
1750 CALL HCHAR(24,8,32,12)
1760 CALL HCHAR(24,20,60)
1770 IF F<>78 THEN 1810
1780 IF SEG$(T$,1,1)="+" THEN 1820
1790 T$=V$
1800 GOTO 1840
1810 IF F<>70 THEN 1840
1820 F=70
1830 T$="+"&SEG$(T$,2,LEN(T$))
1840 M=24
1850 N=8
1860 M$=T$
1870 GOSUB 1680
1880 P=LEN(T$)+8
```

```
1890 CALL HCHAR(24,P,95)
1900 CALL HCHAR(24,P,32)
1910 CALL KEY(3,K,S)
1920 IF S=0 THEN 1890
1930 IF ((K=8)*(P>8)*(F<>70))+((K=8)*(P>9)*(F=70))THEN
      1990
1940 IF (F=78)*(P<20)*(((K>47)*(K<58))+(K=45)+(K=46)+(
     K=69)+(K=43))THEN 2030
1950 IF (F=84)*(P<20)*(K<>13)THEN 2030
1960 IF (F=70)*((P=11)+(P=14)+(P=17))*((K=43)+(K=45)+(
     K=47)+(K=42)+(K=94))THEN 2030
1970 IF (F=70)*(P<20)*(P<>11)*(P<>14)*(P<>17)*(K>64)*(
     K<91)THEN 2030
1980 IF K=13 THEN 2060 ELSE 1890
1990 T$=SEG$(T$,1,LEN(T$)-1)
2000 CALL HCHAR(24,P,32)
2010 CALL HCHAR(24,20,60)
2020 GOTO 1880
2030 T$=T$&CHR$(K)
2040 CALL HCHAR(24,P,K)
2050 GOTO 1880
2060 IF F<>78 THEN 2120
2070 B$(X+D,Y+E)=T$
2080 A$(X+D,Y+E)=""
2090 V$=T$
2100 T$=""
2110 GOTO 2150
2120 A$(X+D,Y+E)=T$
2130 B$(X+D,Y+E)=""
2140 V$=""
2150 Q=Q-LEN(V$)-LEN(T$)+((LEN(V$)>0)*4)+((LEN(T$)>0)*
     4)
2160 M=D+5
2170 N=E*13+5
2180 GOSUB 2220
2190 M$=L$
2200 GOSUB 1680
2210 GOTO 350
2220 L$="              "
2230 IF T$<>"" THEN 2340
2240 IF V$="" THEN 2360
2250 IF POS(V$,"E",1)>1 THEN 2320
2260 IF (VAL(V$)>999999999)+(VAL(V$)<-99999999)THEN 23
     20
2270 V$=STR$(INT(VAL(V$)*100)/100)
2280 IF (LEN(V$)>2)*(LEN(V$)-POS(V$,".",1)=2)THEN 2320
2290 IF (LEN(V$)>1)*(LEN(V$)-POS(V$,".",1)=1)THEN 2310
2300 V$=V$&".0"
2310 V$=V$&"0"
2320 L$=SEG$(L$&V$,LEN(V$)+1,12)
2330 GOTO 2360
2340 IF (SEG$(T$,1,1)="+")*(V$<>"")THEN 2260
2350 L$=SEG$(T$&L$,1,12)
2360 RETURN
2370 END
```

What follows is a more detailed analysis of the *TI CALC* program. See the discussion which follows each section of program code.

```
100 DIM A$(26,26),B$(26,26)
110 X=1
120 Y=1
130 Q=5000
```

The initialization routine (lines 100-130) defines two arrays (A$ and B$) to hold the spreadsheet data, sets the upper left cell to the current cell (X,Y), and sets the number of free bytes available for data (Q). The number must be changed to 4200 if you are using Extended BASIC.

```
140 CALL CLEAR
150 IF Y<26 THEN 180
160 E=1
170 Y=25
180 IF X<15 THEN 210
190 D=X-14
200 X=14
210 PRINT "** TI-99/4A  CALC ** ";STR$(INT(Q/50));"% *
     *";::TAB(8);CHR$(Y+64);TAB(22);CHR$(Y+65);::
220 FOR R=X TO X+12
230 T$=A$(R,Y)
240 V$=B$(R,Y)
250 GOSUB 2220
260 M$=L$
270 T$=A$(R,Y+1)
280 V$=B$(R,Y+1)
290 GOSUB 2220
300 PRINT CHR$(R+64);" ";M$;TAB(16);L$
310 NEXT R
320 PRINT :"G:GOTO     C:CALC":"N:NUMBER  T:TEXT    F:FO
     RMULA":"L:LOAD     S:SAVE    E:END":::
330 K=0
340 GOTO 1520
```

The table print routine (lines 140-340) first determines if the upper left corner must be adjusted (chosen upper left greater than cell NY), prints the amount of free memory (50 must be changed to 42 if using Extended BASIC), calls the number formatting routine, and prints the option information.

```
350 IF X+D<G THEN 370
360 G=X+D
370 IF Y+E<H THEN 390
380 H=Y+E
```

The next routine (lines 350-380) checks to see if a new greater row or column has been used in the spreadsheet. This is an important routine because the smaller the area used, the faster the program runs.

```
390 T$=A$(X+D,Y+E)
400 V$=B$(X+D,Y+E)
410 CALL HCHAR(24,3,32,28)
420 CALL HCHAR(24,3,63)
430 CALL KEY(3,K,S)
440 IF S=0 THEN 430
450 CALL HCHAR(24,4,K)
460 IF (K>7)*(K<12)THEN 1520
470 IF (K=76)+(K=83)+(K=69)THEN 700
480 IF K=71 THEN 510
490 IF K=67 THEN 810
500 IF (K=78)+(K=84)+(K=70)THEN 1720 ELSE 420
```

The option getting routine (lines 390-500) sets temporary strings to the current cells' value (T$,V$), uses the KEY subprogram to get the user's option, makes sure it is valid, and goes to the correct section of the program.

```
510 M=24
520 N=7
530 M$="ROW?"
540 GOSUB 1680
550 CALL KEY(3,K,S)
560 IF (K<65)+(K>90)THEN 550
570 CALL HCHAR(24,12,K)
580 X=K-64
590 M=24
600 N=16
610 M$="COLUMN?"
620 GOSUB 1680
630 CALL KEY(3,K,S)
640 IF (K<65)+(K>90)THEN 630
650 CALL HCHAR(24,24,K)
660 Y=K-64
670 D=0
680 E=0
690 GOTO 140
```

The goto routine (lines 510-690) uses the message printing routine and the KEY subprogram to prompt the user for a new current cell. It then makes sure the choices are valid, sets the new current cell (X,Y), resets the cell displacement (D,E), and returns to the table print routine.

```
700 F=K
710 M=24
720 N=7
730 M$="ARE YOU SURE (Y/N)?"
740 GOSUB 1680
750 CALL KEY(3,K,S)
760 IF K=89 THEN 780
770 IF K=78 THEN 350 ELSE 750
780 CALL CLEAR
790 IF F=69 THEN 2370
800 IF F=76 THEN 1220 ELSE 1400
```

The confirmation routine (lines 700-800) sets a flag (F) to the user's choice, asks for confirmation, and if so, goes to the part of the program as pointed by the flag.

```
810 FOR R=1 TO G
820 FOR C=1 TO H
830 IF SEG$(A$(R,C),1,1)="+" THEN 870
840 NEXT C
850 NEXT R
860 GOTO 140
870 Q=Q+LEN(B$(R,C))-((LEN(B$(R,C))>0)*4)
880 Z=0
890 T$=A$(R,C)
900 FOR L=1 TO LEN(T$)STEP 3
910 T=0
920 O=ASC(SEG$(T$,L,1))
930 I=ASC(SEG$(T$,L+1,1))-64
940 J=ASC(SEG$(T$,L+2,1))-64
950 IF (I<1)+(I>26)+(J<1)+(J>26)+((O<>42)*(O<>43)*(O<>
    45)*(O<>47)*(O<>94))THEN 1150
960 IF B$(I,J)<>"" THEN 970 ELSE 980
970 T=VAL(B$(I,J))
980 IF O=94 THEN 1000
990 ON O-41 GOTO 1020,1040,1150,1060,1150,1080
1000 Z=Z^T
1010 GOTO 1100
1020 Z=Z*T
1030 GOTO 1100
1040 Z=Z+T
1050 GOTO 1100
1060 Z=Z-T
1070 GOTO 1100
1080 IF T=0 THEN 1150
1090 Z=Z/T
1100 NEXT L
1110 IF POS(STR$(Z),"*",1)>1 THEN 1150
1120 B$(R,C)=STR$(Z)
1130 Q=Q-LEN(B$(R,C))+((LEN(B$(R,C))>0)*4)
1140 GOTO 840
1150 Q=Q-LEN(B$(R,C))+((LEN(B$(R,C))>0)*4)
1160 M=24
1170 N=3
1180 M$=" ERROR AT "&CHR$(R+64)&CHR$(C+64)&" - PRESS A
    NY KEY"
1190 GOSUB 1680
1200 CALL KEY(3,K,S)
1210 IF S=0 THEN 1200 ELSE 140
```

The CALC routine (lines 810-1210) has four main parts. First, it loops through the spreadsheet from left to right, top to bottom, and looks for formulas. Second, it concantenates each individual calculation and checks its validity. Third, it performs the mathmatical operation (+,-,*,/,^) to the accumulator variable (Z). Fourth, there is a

short error message printing routine for invalid formulas or division by zero.

```
1220 PRINT "CALC LOAD"
1230 FOR R=1 TO G
1240 FOR C=1 TO H
1250 A$(R,C)=""
1260 B$(R,C)=""
1270 NEXT C
1280 NEXT R
1290 OPEN #1:"CS1",INTERNAL,INPUT ,FIXED 192
1300 INPUT #1:G,H,Q
1310 FOR C=1 TO H
1320 INPUT #1:A$(1,C),A$(2,C),A$(3,C),A$(4,C),A$(5,C),
     A$(6,C),A$(7,C),A$(8,C),A$(9,C),A$(10,C),A$(11,C)
     ,A$(12,C),A$(13,C)
1330 INPUT #1:B$(1,C),B$(2,C),B$(3,C),B$(4,C),B$(5,C),
     B$(6,C),B$(7,C),B$(8,C),B$(9,C),B$(10,C),B$(11,C)
     ,B$(12,C),B$(13,C)
1340 IF G<14 THEN 1370
1350 INPUT #1:A$(14,C),A$(15,C),A$(16,C),A$(17,C),A$(1
     8,C),A$(19,C),A$(20,C),A$(21,C),A$(22,C),A$(23,C)
     ,A$(24,C),A$(25,C),A$(26,C)
1360 INPUT #1:B$(14,C),B$(15,C),B$(16,C),B$(17,C),B$(1
     8,C),B$(19,C),B$(20,C),B$(21,C),B$(22,C),B$(23,C)
     ,B$(24,C),B$(25,C),B$(26,C)
1370 NEXT C
1380 CLOSE #1
1390 GOTO 140
```

The load routine (lines 1220-1390) clears the spreadsheet and loads a cassette data file.

```
1400 PRINT "CALC SAVE"
1410 OPEN #1:"CS1",INTERNAL,OUTPUT,FIXED 192
1420 PRINT #1:G,H,Q
1430 FOR C=1 TO H
1440 PRINT #1:A$(1,C),A$(2,C),A$(3,C),A$(4,C),A$(5,C),
     A$(6,C),A$(7,C),A$(8,C),A$(9,C),A$(10,C),A$(11,C)
     ,A$(12,C),A$(13,C)
1450 PRINT #1:B$(1,C),B$(2,C),B$(3,C),B$(4,C),B$(5,C),
     B$(6,C),B$(7,C),B$(8,C),B$(9,C),B$(10,C),B$(11,C)
     ,B$(12,C),B$(13,C)
1460 IF G<14 THEN 1490
1470 PRINT #1:A$(14,C),A$(15,C),A$(16,C),A$(17,C),A$(1
     8,C),A$(19,C),A$(20,C),A$(21,C),A$(22,C),A$(23,C)
     ,A$(24,C),A$(25,C),A$(26,C)
1480 PRINT #1:B$(14,C),B$(15,C),B$(16,C),B$(17,C),B$(1
     8,C),B$(19,C),B$(20,C),B$(21,C),B$(22,C),B$(23,C)
     ,B$(24,C),B$(25,C),B$(26,C)
1490 NEXT C
1500 CLOSE #1
1510 GOTO 140
```

The save routine (lines 1400-1510) saves the current spreadsheet on cassette for later use.

```
1520 CALL HCHAR(D+5,E*13+4,32)
1530 CALL HCHAR(D+5,E*13+17,32)
1540 IF ((K=8)*(E=1))+((K=9)*(E=0))THEN 1580
1550 IF ((K=8)*(E=0)*(Y>1))+((K=9)*(E=1)*(Y<25))THEN 1
     640
1560 IF ((K=10)*(D<12))+((K=11)*(D>0))THEN 1600
1570 IF ((K=10)*(D=12)*(X<14))+((K=11)*(D=0)*(X>1))THE
     N 1660 ELSE 1610
1580 E=1-E
1590 GOTO 1610
1600 D=D+21-K*2
1610 CALL HCHAR(D+5,E*13+4,62)
1620 CALL HCHAR(D+5,E*13+17,60)
1630 GOTO 350
1640 Y=Y+E*2-1
1650 GOTO 1430
1660 X=X+21-K*2
1670 GOTO 1400
```

The cursor movement routine (lines 1520-1670) checks for a valid movement, determines if a move was made off the screen, changes the current cell, and then either moves the pointing arrows or redraws the entire screen.

```
1680 FOR L=1 TO LEN(M$)
1690 CALL HCHAR(M,N+L-1,ASC(SEG$(M$,L,1)))
1700 NEXT L
1710 RETURN
```

The message printing routine (lines 1680-1710) prints the message (M$) starting at screen row (M) and column (N).

```
1720 F=K
1730 Q=Q+LEN(V$)+LEN(T$)-((LEN(V$)>0)*4)-((LEN(T$)>0)*
     4)
1740 CALL HCHAR(24,7,62)
1750 CALL HCHAR(24,8,32,12)
1760 CALL HCHAR(24,20,60)
1770 IF F<>78 THEN 1810
1780 IF SEG$(T$,1,1)="+" THEN 1820
1790 T$=V$
1800 GOTO 1840
1810 IF F<>70 THEN 1840
1820 F=70
1830 T$="+"&SEG$(T$,2,LEN(T$))
1840 M=24
1850 N=8
1860 M$=T$
1870 GOSUB 1680
1880 P=LEN(T$)+8
1890 CALL HCHAR(24,P,95)
1900 CALL HCHAR(24,P,32)
1910 CALL KEY(3,K,S)
1920 IF S=0 THEN 1890
```

```
1930 IF ((K=8)*(P>8)*(F<>70))+((K=8)*(P>9)*(F=70))THEN
     1990
1940 IF (F=78)*(P<20)*(((K>47)*(K<58))+(K=45)+(K=46)+(
     K=69)+(K=43))THEN 2030
1950 IF (F=84)*(P<20)*(K<>13)THEN 2030
1960 IF (F=70)*((P=11)+(P=14)+(P=17))*((K=43)+(K=45)+(
     K=47)+(K=42)+(K=94))THEN 2030
1970 IF (F=70)*(P<20)*(P<>11)*(P<>14)*(P<>17)*(K>64)*(
     K<91)THEN 2030
1980 IF K=13 THEN 2060 ELSE 1890
1990 T$=SEG$(T$,1,LEN(T$)-1)
2000 CALL HCHAR(24,P,32)
2010 CALL HCHAR(24,20,60)
2020 GOTO 1880
2030 T$=T$&CHR$(K)
2040 CALL HCHAR(24,P,K)
2050 GOTO 1880
2060 IF F<>78 THEN 2120
2070 B$(X+D,Y+E)=T$
2080 A$(X+D,Y+E)=""
2090 V$=T$
2100 T$=""
2110 GOTO 2150
2120 A$(X+D,Y+E)=T$
2130 B$(X+D,Y+E)=""
2140 V$=""
2150 Q=Q-LEN(V$)-LEN(T$)+((LEN(V$)>0)*4)+((LEN(T$)>0)*
     4)
2160 M=D+5
2170 N=E*13+5
2180 GOSUB 2220
2190 M$=L$
2200 GOSUB 1680
2210 GOTO 350
```

The edit routine (lines 1720-2210) sets a flag (F) for the type of edit, adds the current cell memory use (that was previously subtracted) to variable Q, puts the edit arrows on the command line, forces cells with a formula to a formula edit, forces a "+" as the first character of a formula edit, and prints the current cell contents on the edit command line. It then gets the edit one keypress at a time, check its validity according to type of edit and length, does a backspace if <FCTN/S> was pressed and adds a new letter to the current cell string. If <ENTER> was pressed it exits edit command line, changes array contents (A$,B$) of the current cell to new values, subtracts the new memory use for the cell, and calls the message printing routine to put the new value on the screen at the proper location.

```
2220 L$="                "
2230 IF T$<>"" THEN 2340
2240 IF V$="" THEN 2360
2250 IF POS(V$,"E",1)>1 THEN 2320
2260 IF (VAL(V$)>999999999)+(VAL(V$)<-99999999)THEN 23
     20
2270 V$=STR$(INT(VAL(V$)*100)/100)
2280 IF (LEN(V$)>2)*(LEN(V$)-POS(V$,".",1)=2)THEN 2320
2290 IF (LEN(V$)>1)*(LEN(V$)-POS(V$,".",1)=1)THEN 2310
2300 V$=V$&".0"
2310 V$=V$&"0"
2320 L$=SEG$(L$&V$,LEN(V$)+1,12)
2330 GOTO 2360
2340 IF (SEG$(T$,1,1)="+")*(V$<>"")THEN 2260
2350 L$=SEG$(T$&L$,1,12)
2360 RETURN
```

The formatting routine (lines 2220-2360) right justifies numbers and rounds them to two decimal places, right justifies exponential numbers, and left justifies formulas and text.

```
2370 END
```

The last line (line 2370) is the program exit.

SAMS

TI-99/4A™ Calc

Use your TI-99/4A personal computer to learn why spreadsheet programs are so useful and popular. *TI-99/4A Calc* shows you how spreadsheets work and how to use them most effectively.

- Includes a complete 676 cell spreadsheet program
- Runs on any TI-99/4A computer with cassette recorder
- Permits loading, editing, and saving of individual spreadsheets
- Calculations can include numbers, formulas, and cell referencing
- Provides six spreadsheets already saved on cassette to help you

Machine Requirements:

- **TI-99/4A Personal Computer**
- **Cassette Player**

Howard W. Sams & Co., Inc.
4300 West 62nd Street, Indianapolis, Indiana 46268 U.S.A.

$8.95/22414

ISBN: 0-672-22414-3