Model 2000

BASIC

TANDY
TRS-80
COMPUTER PRODUCTS

TRS-80

Running GRAPHICS.BAS:

A Demonstration of High Resolution Graphics

We have added a file, GRAPHICS.BAS, to your MS-DOS system diskette. This program demonstrates some of the high resolution graphics available with BASIC. To use the program, you need the CM-1 Color Monitor and a Monochrome Graphics Option board upgraded with the Color Graphics Option kit.

Note: If you have a VM-1 Monochrome Monitor and the Monochrome Graphics Option Board, you can run the program after removing the color-related statements and changing the SCREEN 3 statements to SCREEN 4. (See *Model 2000 BASIC Reference* for more information on these statements.)

If you do not plan to use GRAPHICS.BAS, you may delete the file from the diskette, as described in Chapter 8 of *Introduction to the Model 2000*. Otherwise, run the file by typing (at the system prompt):

BASIC GRAPHICS.BAS (ENTER)

When finished, return to MS-DOS by typing:

SYSTEM (ENTER)

Tandy Corporation
Part No. 8759267

Cat. No. 26-5103

BASIC

TERMS AND CONDITIONS OF SALE AND LICENSE OF RADIO SHACK AND TANDY COMPUTER EQUIPMENT AND SOFTWARE PURCHASED FROM A RADIO SHACK COMPANY-OWNED COMPUTER CENTER, RETAIL STORE OR FROM A RADIO SHACK FRANCHISEE OR DEALER AT ITS AUTHORIZED LOCATION

LIMITED WARRANTY

I. CUSTOMER OBLIGATIONS

A. CUSTOMER assumes full responsibility that this computer hardware purchased (the "Equipment") and any copies of software included with the Equipment or licensed separately (the "Software") meets the specifications, capacity, capabilities, versatility, and other requirements of CUSTOMER.

B. CUSTOMER assumes full responsibility for the condition and effectiveness of the operating environment in which the Equipment and Software are to function, and for its installation.

II. RADIO SHACK LIMITED WARRANTIES AND CONDITIONS OF SALE

A. For a period of ninety (90) calendar days from the date of the Radio Shack sales document received upon purchase of the Equipment, RADIO SHACK warrants to the original CUSTOMER that the Equipment and the medium upon which the Software is stored is free from manufacturing defects. **This warranty is only applicable to purchases of Radio Shack and Tandy Equipment by the original customer from Radio Shack company-owned computer centers, retail stores and from Radio Shack franchisees and dealers at its authorized location.** The warranty is void if the Equipment's case or cabinet has been opened, or if the Equipment or Software has been subjected to improper or abnormal use. If a manufacturing defect is discovered during the stated warranty period, the defective Equipment must be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer for repair, along with a copy of the sales document or lease agreement. The original CUSTOMER'S sole and exclusive remedy in the event of a defect is limited to the correction of the defect by repair, replacement, or refund of the purchase price, at RADIO SHACK'S election and sole expense. RADIO SHACK has no obligation to replace or repair expendable items.

B. RADIO SHACK makes no warranty as to the design, capability, capacity, or suitability for use of the Software, except as provided in this paragraph. Software is licensed on an "AS IS" basis, without warranty. The original CUSTOMER'S exclusive remedy, in the event of a Software manufacturing defect, is its repair or replacement within thirty (30) calendar days of the date of the Radio Shack sales document received upon license of the Software. The defective Software shall be returned to a Radio Shack Computer Center, a Radio Shack retail store, participating Radio Shack franchisee or Radio Shack dealer along with the sales document.

C. Except as provided herein, no employee, agent, franchisee, dealer or other person is authorized to give any warranties of any nature on behalf of RADIO SHACK.

D. **Except as provided herein, Radio Shack makes no express warranties, and any implied warranty of merchantability or fitness for a particular purpose is limited in its duration to the duration of the written limited warranties set forth herein.**

E. Some states do not allow limitations on how long an implied warranty lasts, so the above limitation(s) may not apply to CUSTOMER.

III. LIMITATION OF LIABILITY

A. **Except as provided herein, Radio Shack shall have no liability or responsibility to customer or any other person or entity with respect to any liability, loss or damage caused or alleged to be caused directly or indirectly by "Equipment" or "Software" sold, leased, licensed or furnished by Radio Shack, including, but not limited to, any interruption of service, loss of business or anticipatory profits or consequential damages resulting from the use or operation of the "Equipment" or "Software". In no event shall Radio Shack be liable for loss of profits, or any indirect, special, or consequential damages arising out of any breach of this warranty or in any manner arising out of or connected with the sale, lease, license, use or anticipated use of the "Equipment" or "Software".**

Notwithstanding the above limitations and warranties, Radio Shack's liability hereunder for damages incurred by customer or others shall not exceed the amount paid by customer for the particular "Equipment" or "Software" involved.

B. RADIO SHACK shall not be liable for any damages caused by delay in delivering or furnishing Equipment and or Software.

C. No action arising out of any claimed breach of this Warranty or transactions under this Warranty may be brought more than two (2) years after the cause of action has accrued or more than four (4) years after the date of the Radio Shack sales document for the Equipment or Software, whichever first occurs.

D. Some states do not allow the limitation or exclusion of incidental or consequential damages, so the above limitation(s) or exclusion(s) may not apply to CUSTOMER.

IV. RADIO SHACK SOFTWARE LICENSE

RADIO SHACK grants to CUSTOMER a non-exclusive, paid-up license to use the RADIO SHACK Software on **one** computer, subject to the following provisions:

A. Except as otherwise provided in this Software License, applicable copyright laws shall apply to the Software.

B. Title to the medium on which the Software is recorded (cassette and or diskette) or stored (ROM) is transferred to CUSTOMER, but not title to the Software.

C. CUSTOMER may use Software on one host computer and access that Software through one or more terminals if the Software permits this function.

D. CUSTOMER shall not use, make, manufacture, or reproduce copies of Software except for use on **one** computer and as is specifically provided in this Software License. Customer is expressly prohibited from disassembling the Software.

E. CUSTOMER is permitted to make additional copies of the Software **only** for backup or archival purposes or if additional copies are required in the operation of **one** computer with the Software, but only to the extent the Software allows a backup copy to be made. However, for TRSDOS Software, CUSTOMER is permitted to make a limited number of additional copies for CUSTOMER'S own use.

F. CUSTOMER may resell or distribute unmodified copies of the Software provided CUSTOMER has purchased one copy of the Software for each one sold or distributed. The provisions of this Software License shall also be applicable to third parties receiving copies of the Software from CUSTOMER.

G. All copyright notices shall be retained on all copies of the Software.

V. APPLICABILITY OF WARRANTY

A. The terms and conditions of this Warranty are applicable as between RADIO SHACK and CUSTOMER to either a sale of the Equipment and or Software License to CUSTOMER or to a transaction whereby RADIO SHACK sells or conveys such Equipment to a third party for lease to CUSTOMER.

B. The limitations of liability and Warranty provisions herein shall inure to the benefit of RADIO SHACK, the author, owner and or licensor of the Software and any manufacturer of the Equipment sold by RADIO SHACK.

VI. STATE LAW RIGHTS

The warranties granted herein give the **original** CUSTOMER specific legal rights, and the **original** CUSTOMER may have other rights which vary from state to state.

10 9 8 7 6 5 4 3

Introduction

This manual is about the popular GW-BASIC from Microsoft. BASIC for MS-DOS is an "interpreter." When you run a program, it executes each statement one at a time. This makes it quick and easy to use. It also allows you to take advantage of many MS-DOS features, such as:

- Faster running programs
- Expanded graphics capabilities

About this Manual

This is a reference manual, not a tutorial. We assume you already know BASIC and are using this manual to quickly find the information you need. If you do not know BASIC, many excellent books are available at your local bookstore written in a tutorial fashion to teach you BASIC.

Section I — Operations. This section shows how to load BASIC. It also demonstrates how to write, run, and save a BASIC program on disk.

Section II — The BASIC Language. This section includes a definition for each BASIC keyword (statements and functions) in alphabetical order. In addition, it shows how to write a program to store data on disk.

Notations

CAPITALS	material that must be entered exactly as it appears.
italics	words, letters, characters, or values you must supply from a set of acceptable entries.
. . . (ellipsis)	items preceding the ellipsis may be repeated.
X'NNNN	NNNN is a hexadecimal number.
O'NNNN	NNNN is an octal number.

(KEYNAME)	a key on your keyboard.
␣	a blank space character (ASCII code 32). For example, in BASIC␣␣PROG two spaces are between BASIC and PROG.

Terms

buffer	a number in the range 1 to 15. This refers to an area in memory that BASIC uses to create and access a disk file. Once you use a buffer to create a file, you cannot use it to create or access any other files; you must first close the file. You may only access an open file with the buffer used to open it.
[*parameters*]	information you supply to specify how a command is to operate. Parameters enclosed in brackets are optional.
[*arguments*]	expressions you supply for a function to evaluate. Arguments enclosed in brackets are optional.
syntax	a command with its parameter(s), or a function with its argument(s). This shows the format to use for entering a keyword in a program line.
line	a numeric expression that identifies a BASIC program line. Each line has a number between 0 and 65529.
integer	any integer expression. It may consist of an integer or of several integers joined by operators. Integers are whole numbers between −32768 and 32767.

string	any string expression. It may consist of a string, or of several strings joined by operators. A string is a sequence of characters that is to be taken verbatim.
number	any numeric expression. It may consist of a number or of several numbers joined by operators.
dummy number or *dummy string*	a number (or string) used in an expression to meet syntactic requirements, but the value of which is insignificant.

Table of Contents

Using BASIC

Section I

Operations

Chapter 1

Sample Session

The easiest way to learn how BASIC operates is to write and run a program. This chapter provides sample statements and instructions to help familiarize you with the way BASIC works.

The main steps in running a program are:

A) Loading BASIC
B) Typing the program
C) Editing the program
D) Running the program
E) Saving the program on disk
F) Loading the program back into memory

Loading BASIC

We recommend that you read your *Introduction to Model 2000* for complete startup information on your Model 2000 before you load BASIC. It details the necessary steps to get to the MS-DOS command level prompt.

At the MS-DOS system prompt A>, you can load BASIC into the computer's memory by typing

BASIC (ENTER)

A paragraph with copyright information appears on your screen, followed by: Ok

You may now begin using BASIC.

Options for Loading BASIC

When loading BASIC, you can also specify a set of options. They are:

BASIC [*filename*] [/F:*number of files*] [/C:*buffer size*] [/M:*highest memory location*] [/S:*record length*]

Filename specifies a program to run immediately after BASIC is started.

/F: specifies the maximum number of data files that may be open at any one time (from 0-15). If you omit this option, the number of files defaults to three.

Each file you specify may use up to 190 bytes of memory. Sequential access files always use 190 bytes of memory. The amount of memory a direct access file uses depends on the record size set with the /S: option. Each file uses 62 bytes of memory for the file control block, plus the record size. For example, if you specify a record size of 50 with the /S: switch, the file uses 112 bytes.

/C: specifies the size of the receive buffer for RS232 communication. If you omit the /C: option, BASIC allocates 256 bytes for the receive buffer. The transmit buffer is always 128 bytes.

/M: specifies the highest memory location for BASIC to use. Omit this option unless you plan to call assembly-language subroutines. (In that case, you may want to set the highest memory location well below the top of BASIC's data segment.) If you omit this option, the system allocates 64K bytes of memory to BASIC.

/S: specifies the maximum record size for direct access files. If you omit the /S: option, BASIC assumes 128 bytes.

Examples

A>
BASIC PAYROLL /F:5 (ENTER)

initializes BASIC, then loads and runs the program PAYROLL; allows five data files to be open; uses all memory available.

A>
BASIC /M:21000 (ENTER)

initializes BASIC; allows three data files to be open; sets the highest memory location to be used by BASIC at 21000, the first 21K bytes of BASIC's data segment.

A>
BASIC /M:21000 /F:6 (ENTER)

initializes BASIC; sets the highest memory location at 21000; allows six data files to be open. Notice that the sequence in which the /M: and /F: options are specified is irrelevant.

A>
BASIC

initializes BASIC; allows three data files to be open; uses all memory available.

Typing the Program

Let's write a small BASIC program. Before pressing (ENTER) after each line, check the spelling. If you have made any mistakes, use the ← key to correct them.

10 A$ = "WILLIAM SHAKESPEARE WROTE " (ENTER)
15 B$ = "THE MERCHANT OF VENICE" (ENTER)
20 PRINT A$; B$ (ENTER)

Check your spelling again. If it is still not perfect, enter the line number where you made the mistake. Then type the entire line again.

For example, suppose you had typed:

```
15 B$ = "THE VERCHANT OF VENICE"
```

To correct line 15, retype it:

```
15 B$ = "THE MERCHANT OF VENICE" (ENTER)
```

Then type:

```
RUN (ENTER)
```

Your screen should display:

```
WILLIAM SHAKESPEARE WROTE THE MERCHANT
OF VENICE
```

BASIC replaced line 15 in the original program with the most recent line 15.

> **Note:** BASIC "reads" your program lines in numerical order. It doesn't matter if you entered line 15 after line 20; BASIC still reads and executes 15 before "looking" at 20.

BASIC has a powerful set of commands that allow you to correct mistakes without retyping the entire line. These commands are discussed in Chapter 3, the "Line Edit Mode."

Saving the Program on Disk

You can save any BASIC program on disk. To do this, you assign it a *filespec*. The filespec tells BASIC on which disk you want to save the file and the name of the file. A filespec consists of drive identifier, filename, and extension. Only the filename is required. The filespec must be enclosed in double quotes.

Filenames must conform to the MS-DOS file naming conventions. A filename can have a maximum of eight alphanumeric characters. The first character must be a letter, A through Z. The remaining seven characters may be any of the following:

the letters A through Z
the digits 0 through 9
the special characters <, >, (,), {, }, @, #, $, %, ^, &, !, -, _, ', and /

The extension may be up to three characters long and must also begin with a letter. The other two characters may be any of the characters that are allowed in the filename. A period (.) must be included between the filename and the extension. If you omit an extension with the SAVE, LOAD, MERGE, and RUN commands, BASIC appends the extension **.BAS**.

The drive identifier specifies on which disk you want BASIC to save your program. The drive identifier precedes the filename and extension and may be any of the valid drive letters (A through D); it must be followed by a colon. If you omit the drive identifier, BASIC saves the file on the MS-DOS current drive.

For example, to save the program we just wrote on Drive B, assign it the filename "AUTHOR.BAS". Type the following command:

SAVE "B:AUTHOR.BAS" (ENTER)

It takes a few seconds for the computer to find a place on disk to store a program and to copy the program from memory to the disk. When the program is saved on the disk, the screen displays Ok.

Examples

SAVE "AUTHOR.WIL" (ENTER)

saves the program under the filename AUTHOR, with the extension .WIL on the MS-DOS current drive.

SAVE "A:AUTHOR" (ENTER)

saves the program under the filename "AUTHOR" on the disk in Drive A. Because no extension is given, BASIC appends the extension .BAS.

Loading the Program

If, after writing or running other programs, you want to use this program again, you must "load" it back into memory. To do this, type:

LOAD "*filespec*", R

Example

LOAD "AUTHOR", R (ENTER)

tells the computer to load the program "AUTHOR" from disk into memory; option R tells the computer to run it.

Another way to load and run a program is to type:

RUN "*filespec*"

RUN automatically loads and runs the program specified by "*filespec*".

The SAVE, LOAD, and RUN commands are discussed in more detail in Chapter 7.

Chapter 2

Command And Execution Modes

This chapter describes BASIC's command and execution modes. BASIC is in the command mode when you are typing in program lines and immediate lines. BASIC is in the execution mode when it is performing the instructions in the program and immediate lines.

Command Mode

Whenever you enter the command mode, BASIC displays the prompt:

OK

In the command mode, BASIC does not "read" your input until you complete a "logical line" by pressing (ENTER). This is called "line input," as opposed to "character input."

A logical line is a string of up to 255 characters and is always terminated by pressing (ENTER). Of these 255 characters, 249 are reserved for the line itself; the other six are reserved for the line number and the space following the line number.

A physical line, on the other hand, is one line on the display. It contains a maximum of 80 characters.

For example, if you type 100 R's and then press (ENTER), you have two physical lines, but only one logical line.

Interpretation of a Line

BASIC always ignores leading spaces in the line — it jumps ahead to the first non space character. If this character is not a digit, BASIC treats the line as an immediate line. If it is a digit, BASIC treats the line as a program line.

For example, if you type:

```
PRINT "THE TIME IS" TIME$ (ENTER)
```

BASIC takes this as an immediate line.

But if you type:

10 PRINT "THE TIME IS" TIME$ (ENTER)

BASIC takes this as a program line.

Immediate Lines

An immediate line consists of one or more statements separated by colons. There is no line number in an immediate line. The line is executed as soon as you press (ENTER). After BASIC executes the line, it is no longer in memory. The values of variables and constants are still in memory, but the statement no longer exists. Immediate lines are useful for using the computer as a calculator for quick computations that don't require an entire program. For example:

```
Ok
MILES = 133:GAS = 11:MPG = MILES/GAS
```

After this statement is executed, the value of the variables MILES, GAS, and MPG still exist in memory. But the instruction itself does not.

```
Ok
CLS: PRINT "THE SQUARE ROOT OF 2 IS" SQR(2)
```

is an immediate line. When you press (ENTER), BASIC executes it.

Program Lines

A program line consists of a line number in the range 0 to 65529, followed by one or more statements separated by colons. When you press (ENTER), the line is stored in memory. All lines that you enter with a line number are stored in memory until you execute a RUN or other execute command. For example:

100 CLS: PRINT "THE SQUARE ROOT OF 2 IS" SQR(2)

is a program line. When you press (ENTER), BASIC stores it in memory. To execute it, type:

RUN (ENTER)

Note: If you include numeric constants in a line, BASIC evaluates them as soon as you press (ENTER); it does not wait until you RUN the program. If any numbers are out of range for their type, BASIC returns an error message immediately after you press (ENTER).

Special Keys

The CTRL and ALT keys have special functions in BASIC. When you press and hold one of these keys while typing another key, BASIC performs functions to make entering and editing program lines and immediate lines easier. These keys have no function in BASIC unless they are pressed with another key. For example, if you press (CTRL) key and type (H), BASIC backspaces and deletes the character.

Special Keys in the Command Mode

(BACKSPACE) or (CTRL)(H)	Backspaces the cursor, erasing the preceding character in the line. Use this to correct typing errors before pressing (ENTER).
(SPACE BAR)	Enters a blank space character and advances the cursor.
(BREAK) or (CTRL)(C)	Interrupts line entry and starts over with a new line.
(CTRL)(J)	Line feed — starts a new physical line without ending the current logical line.
(CAPS)	Switches the display to either all uppercase or uppercase/lowercase mode.
(ENTER) or (CTRL)(M)	Ends the current logical line. BASIC "takes" the line.

CTRL U or ESC	Erases the current line.
CTRL ← or ←	Moves the cursor one position to the left.
CTRL \ or →	Moves the cursor one position to the right.
CTRL F or CTRL →	Moves the cursor to the first character in the next word to the right of the current cursor positon.
CTRL B or CTRL ←	Moves the cursor to the first character in the word to the left of the current cursor position.
CTRL R or INSERT	Turns on insert mode if it is off; turns off insert mode if it is on.
DELETE	Deletes the character at the current cursor position.
CTRL W	Deletes the next word to the right of the cursor.
CTRL T	Displays the soft key values of the 12 Function Keys.
END or CTRL N	Moves the cursor to the last character in the logical line.
CTRL END or CTRL E	Deletes all characters from the current cursor position to the end of the line.
CTRL I or TAB	Advances the cursor to the next tab position. Tab positions are set at every eight character positions.
CTRL G	Rings the bell at the terminal.

Some BASIC keywords are associated with alphabetic characters (A-Z). To enter these keywords easily, press (ALT) and the corresponding letter. BASIC inserts the keyword at the current cursor position. The keywords and their associated letters are listed below. Letters that don't have an associated keyword are indicated by "(none)."

A	AUTO	N	NEXT
B	BSAVE	O	OPEN
C	COLOR	P	PRINT
D	DELETE	Q	(none)
E	ELSE	R	RUN
F	FOR	S	SCREEN
G	GOTO	T	THEN
H	HEX$	U	USING
I	INPUT	V	VAL
J	(none)	W	WIDTH
K	KEY	X	XOR
L	LOCATE	Y	(none)
M	MOTOR*	Z	(none)

*MOTOR is a reserved word, but is not a recognized statement in this implementation of BASIC.

Execution Mode

When BASIC is executing statements (immediate lines or programs), it is in the execution mode. In this mode, the contents of the video display are under program control.

Special Keys in the Execution Mode

(HOLD) or (CTRL)(S)	Pauses execution. Press any other key (except (BREAK)) to continue.
(BREAK)	Terminates execution and returns you to command mode.
(ENTER) or (CTRL)(M)	Interprets data entered from the keyboard as a response to the INPUT statement.

Chapter 3

The Line Editor

Your BASIC Editor lets you "debug" (correct errors in) your BASIC program quickly and efficiently without retyping entire lines.

To enter line edit mode type :

EDIT *line number* (ENTER)

This lets you edit the specified line number. (If the line number you specify has not been used, an "Undefined line number" error occurs.)

You may also use the LIST command to list one or several lines before you make the changes. If you LIST one line you can use the keys described in the next section to make changes to the line. If you LIST several lines, see the last section of this chapter, "Changing Lines Anywhere on the Screen."

You may also type:

EDIT . (ENTER)

The period after EDIT means that you want to edit the current program line, the last line entered, the last line altered, or a line in which an error has occurred. Notice that you need to type a blank before the period; otherwise, BASIC gives you a "Syntax error" message.

For example, type the following line and press (ENTER).

100 PRINT "This is our example line."

This line will be used in exercising all the edit subcommands described below.

Now type EDIT 100 and press (ENTER). BASIC displays the entire line and positions the cursor under the first digit of the line number. This starts the editor. You may now begin editing line 100. Line 100 can be modified by using any of the special keys described below. **Note:** None of the changes you make to a program line are entered until you press (ENTER).

Special Keys in the Edit Mode

(ENTER) or (CTRL)(M)	Records the changes you made in the current line and returns you to the command mode.
← or (CTRL)(])	Moves the cursor one position to the left. If you advance the cursor past the left-hand margin of the screen, it moves to the right-hand margin of the screen on the previous line.
→ or (CTRL)(\)	Moves the cursor one position to the right. If you advance the cursor past the right-hand margin of the screen, it moves to the left-hand margin of the screen on the next line.
(SPACEBAR)	Changes the character to a blank and advances the cursor one position to the right.
(CTRL)(→) or (CTRL)(F)	Moves the cursor right to the next word. The next word is the next letter or number that follows a blank or a special character.
(CTRL)(←) or (CTRL)(B)	Moves the cursor to the first character in the previous word. The previous word is the next letter or number to the left of the cursor that precedes a blank or special character.
(END) or (CTRL)(N)	Moves the cursor to the last character in the logical line.

DELETE	Erases the character at the current cursor position. All characters to the right of the cursor and subsequent characters and lines within the logical line move left or up one position.
BACKSPACE or CTRL H	Erases the character to the left of the cursor. All characters to the right and subsequent characters and lines within the logical line move left or up one position.
CTRL END or CTRL E	Erases all characters from the current cursor position to the end of the logical line.
CTRL U or ESC	Erases the entire logical line from the screen. BASIC does not record in memory any of the changes made to the line. You may press either of these anywhere in the line to cancel changes made.
TAB or CTRL I	Moves the cursor to the next tab position. Tab positions are set at every eight positions. As the cursor advances to the next tab position, the characters in the positions it is tabbing over are printed.
BREAK or CTRL C	Returns to direct mode and does not record in memory any of the changes to the line currently being edited.
CTRL K or HOME	Moves the cursor to the first position in row one.
CTRL L	Clears the screen and positions the cursor at the first position in row one.

(INSERT)	Allows you to enter characters between other characters that are already in the line. To insert characters press (INSERT) and type the characters you want to insert. All other characters on the logical line move to the right or down each time you insert a character. If while you are inserting characters you press the (TAB) key, blanks are inserted from the current cursor position to the next tab position. After you insert all the characters, press (INSERT) again and continue editing the line.
(CTRL)(Z)	Clears the screen from the current cursor position to the end of the screen.

Sample Session

Type

```
EDIT 100
```

Use the right arrow to space across the line to the "T" in "This." Type lowercase "t" and then (ENTER).

Type

```
LIST 100
```

to see that BASIC stored your change in memory.

Use the edit command to edit the line again. Press (END) to position the cursor on the second set of double quotes. Press (INSERT) and type

```
We inserted the second sentence. (ENTER)
```

Use the list command to see the new statement that is stored in memory.

Type

EDIT . (ENTER)

to edit the line again. You may use the period (.) instead of the line number to edit the current line. Use (TAB) and (→) to position the cursor on the "i" in inserted. Press (CTRL)(END). BASIC deletes all the characters you inserted except "we" and the blank. Use (BACKSPACE) to delete the blank. Use (CTRL)(←) to position the cursor on the previous word. Press (DELETE) twice to delete "we." Press (INSERT), then (") to put the double quote in at the end of the statement. Use the list command to see the new line.

Now let's add another line to the program. Type

200 GOTO 100 (ENTER)
RUN

BASIC is in a loop printing your message repeatedly on the screen. Press (BREAK) to stop program execution.

Type

DELETE 200 (ENTER)

Line 200 is erased from memory.

Use the edit command to edit the line again. Use (→) to position the cursor on the "P" in PRINT. Press (SPACEBAR) to change the "P" to a blank. Press (CTRL)(→) to position the cursor on the "t" in "this." Press (T) to change the lowercase "t" to a capital "T." Instead of pressing (ENTER) after you make the changes, press (ESC). Use the list command to see that BASIC did not record your change, because you pressed (ESC) instead of (ENTER).

Now you have used all the special keys in line edit mode. If you still don't feel comfortable with them, go through the sample session again. If you feel confident that you understand the line editor, read on to learn about some special keys that make it easier and faster to change lines anywhere on the screen.

Changing Lines Anywhere on the Screen

When more than one line is displayed on the screen, you may use the arrow keys to move the cursor around the screen to different program lines to correct errors. After you make all the corrections, you must go to the beginning of each line that you modified and press (ENTER) to record the change in memory. Using the arrow keys to make corrections can be much quicker than typing EDIT and a line number for every line that needs to be changed.

(CTRL)(^) or (↑) — Moves the cursor up one row to the character above the current cursor position.

(CTRL)(–) or (↓) — Moves the cursor down one row to the character below the current cursor position.

You may also use the left and right arrow keys as previously described under "Special Keys in the Edit Mode."

Sample Session

After you type each of the following lines, press (ENTER)

```
10 PRINT "With the fising cost of fuel, ras mileage ";
20 PRINT "has vecome an important donsideration"
30 PRINT "in the purchase of anew vehiclee"
```

Now you can use the arrow keys to correct the mistakes in the program statements.

1. Use (↑), then (→) to position the cursor on the "f" in "fising" in Line 10. Type "r" over the "f."
2. Use (TAB), then (→) to position the cursor on the "r" in "ras." Type (g) over the "r."
3. Use (↓), then (←) to position the cursor over the "d" in "donsideration." Type (c) over the "d."
4. Use (CTRL)(←), then (←) to position the cursor over the "v" in "vecome." Type (b) over the "v."

5. Use ↑, CTRL →, and → to position the cursor over the "n" in "anew". Press INSERT and SPACEBAR to insert a blank between "a" and "new."
6. Use → to space across to the last "e" in "vehiclee." Press DELETE to erase the extra "e".
7. Use ← to position the cursor over the "3" in Line 30. Press ENTER.
8. Use ↑ to position the cursor over the "2" in Line 20. Press ENTER.
9. Use ↑ to position the cursor over the "1" in Line 10. Press ENTER.

More on Line Edit Mode

If your computer encounters a syntax error while executing a program, BASIC automatically enters EDIT. It displays the line that contains the error. For example, type

```
10 A = 2$12 ENTER
RUN
```

The screen displays:

```
?Syntax error in 10
10 A = 2$12
```

EDIT positions the cursor under the first digit of the line number. Now press → to move the cursor to the dollar sign ($) and press DELETE and then ENTER. BASIC stores the corrected line in memory.