MASTERING

The

HP-27S

Scientific Calculator

by Richard E. Harvey

Mastering The HP-27S

SCIENTIFIC CALCULATOR

by Richard E. Harvey

Notice

 June 1988; version 1.0

Other books and software for Hewlett–Packard calculators and computers are available. Please contact the author for further information.

Richard E. Harvey
P.O. Box 5695
Glendale, Arizona 85312 USA

Acknowledgment

The author would like to thank EduCALC Mail Store, and especially Jim Carter, for continued support.

Contents

Introduction

Your HP–27S Scientific Calculator is a part of a new generation of Hewlett–Packard calculator: competent yet simple to use. Much of this ease of use stems from how it works with numbers: The 27 is an Algebraic calculator. The significance of this is that RPN (Reverse Polish Notation) calculators have been a standard tool of HP aficionados for a generation; only since 1987 have we seen a Hewlett–Packard calculator with an equals key.

Perhaps you work with a computer or terminal on the job, maybe a different machine at home, a multi–programmable digital watch, and probably a recalcitrant drive–up bank teller machine. These and any number of other devices were designed by geniuses who have "seen the light" of how you want to work with *their* machine. Unfortunately each of those geniuses has seen a different light; each has a different quest. That's all you need is yet another way to do similar things.

In a refreshingly new way, the folks at HP have recognized that we may not want to have to plow through the Owner's Manual each time we pick up the calculator. The parts that are familiar are, well, familiar; the powerful parts use a menu to refresh your memory.

From there they added sophisticated features like statistics, financial functions, and the highly innovative Solver, each with an extensive library of functions. And modes and cursors; a whole new way to look at calculations.

The 27 is more powerful than personal computers were just a few years ago. And this power is more intuitive and accessible than were those earlier machines. Like setting a digital watch in a leap year, getting the most from your HP–27 takes insight and some experimenting.

It took Hewlett–Packard to find the true power of the Algebraic calculator.

Other Products

Other books and software for Hewlett–Packard calculators and computers are available. One book, "The BASIC HP–71," offers an overview of operating the HP–71 portable computer and programming in BASIC and Assembly Language. And WorkBook71 is a software package including Virtual Memory Spreadsheet, File Manager, Data Format Converter, Full Screen Text Editor and Text Formatter. The ROM version includes an RPN calculator and several other utilities. Please contact the author or the dealer from whom you purchased this book for information.

Richard E. Harvey
P.O. Box 5695
Glendale, Arizona 85312 USA

June 1988; Version 1.0

How to Use This Book

The HP–27S works with formulas in two ways: Keyboard calculations and Solver equations. These are separate and mostly unrelated jobs, so we'll cover them in separate chapters.

We use the text editor to type printed notes, appointment note fields, and for editing Solver equations. Since it's much the same editor regardless of where it's being used, it's explained in a separate chapter; differences in use between applications are noted where appropriate. The text editor is the backbone of Solver; you'll save time and hassles if you familiarize yourself with the text editor before tackling Solver.

Much of this book is reference style, it isn't necessary to read it from start to finish. In fact, the final section is entirely reference: an index (mostly alphabetical) of calculator and Solver commands. After you're long familiar with the workings of your 27, we hope you'll find this listing useful.

At the end of most sections, there are reviews of salient issues, called "Notes about...". If after reviewing these points an item or two seems vague, feel free to pop back to the appropriate headline for another round.

While we'll cover quite a bit of information quickly, there are no prizes for speed reading. Nor are there quizes or grades. Take your time and explore your 27, you may be surprised about how much you'll find.

How ever you decide to use this book, be sure you have your calculator at your side to try things. Don't limit yourself to mimicking the keystrokes in the examples (we've already shown what those keystrokes do). Experiment! Try things, whether you think they'll work or not.

Symbols and Conventions

Example keystrokes are printed in a keyboard-like style; for example [INPUT] or [▲]. The blue shift key is printed as [■]. When used with the shift key, most keys take on a second meaning which is labeled in blue on the face of the keyboard. We'll print the shifted function as though that were what the key said. [SIN] when combined with the shift key gives [■][ASIN]. That's a bit confusing, but less so than demonstrating [■][SIN] and asking you to look at the keyboard to see what it really means.

Keystrokes corresponding to menus assigned to the top row of [△] keys are not printed in the black-on-white style printed in the Owner's Manual because that can be downright difficult to read. Instead we'll use a simpler style: ≡MORE≡.

Following Keystroke Examples

For example, to do calculations with time functions, press [■][MAIN] to exit from any active menu, then press [■][TIME] to enter the TIME menu. Then from the top row, select the [△] key labeled ≡CALC≡ on the menu, and you're in the appropriate sub-menu.

Keystrokes	Reaction
[■][MAIN]	Exit from the active menu. This erases the menu labels from the display.
[■][TIME]	Enter the TIME menu.
≡CALC≡	Now you're in the time calculations sub-menu, ready to do time math.

The calculator display is represented as a box with double lines drawn around it. A single line is drawn around menu keys on the display. When it's likely that there's a flashing cursor present, we'll simulate one with "▓".

Often we'll use a series of these boxes to show step-by-step what happens during an exercise. On the left of the page are the keys to press, the calculator display is in the middle, and a remark or two about what's going on is printed on the right. This illustration shows the [■][PARTS] menu keys on the bottom of the display.

Keystrokes	Display	Remark
[■] [PARTS]	0.41421356237 IP | FP | RND | ABS	The parts menu.

When only a single line is important (such as when no menu keys are used), a single line is shown; this is similar to the format of the Owner's Manual.

Information in examples may overflow onto several lines, especially when dealing with Solver. Instead of printing three little dots (called an ellipsis) to show you that there's more than will fit in the display, we'll print it.

Since an example may be at the end of a discussion, text which follows will not necessarily have anything to do with it.

Hard Copy

This brings us to a reality of the computer age: it's easy to erase things accidentally. Sometime during your partnership with your 27, you will most likely lose some information. Maintaining an up-to-date printed listing of important data and formulas is advisable. If it was worth your time to enter the information, it's worth your time to protect it.

The wireless HP82240A printer is a laudable investment and will pay for itself in time saved. For long term storage, a reasonable precaution is to photocopy the printed listings, because the thermal printer "ink" isn't permanent. Be sure to use black-printing thermal paper because blue thermal paper images reproduce poorly. At the very least, when you store the printed strips in a notebook, be sure that tape doesn't cover the printed image because adhesive tape may cause the image to fade.

If your printer starts churning out reams of unwanted information, turn it off then press a calculator key to halt the 27 from trying to send more data to the printer. This can also happen if you have more than one calculator attempting to use the printer at the same time. Once your printer is off, un-printed infrared data merely floats off to erode the ozone layer.

Actually, these infrared signals are weaker than those emitted by the wireless remote control on your television, and dissipate a couple of feet from the calculator. It's unlikely that the infrared signals will affect any other infrared receiving device you own, nor will the television remote control affect your printer. That's one case where a lack of industry standardization works to our advantage.

Terminology; LCD, CPU and all

Calculators are reflections of the personal beliefs and styles of their creators and users. As such, new calculator words are coined and meanings for existing words may have nothing to do with the original word. When new words are needed, sometimes we end up with *mnemonic* or *octal* or something based on an obscure reference to "Alice In Wonderland."

There were *plugs* and *distributors* before there were automobiles, just as there were *bites* and *operators* before there were computers. When you stop to think about it, it doesn't really matter what you call things, just so you consistently call them the same thing. They're just words.

Occasionally there's a shortage of words. A case in point is the word *display*; we *display* numbers in a *display* format on the calculator *display*.

The Owner's Manual avoids the terms *computer* and *program*. Well, we'll use those terms right up front; the 27 erases that fine line between calculator and computer. Solver is programming; on a high level, but programming. There's nothing truly difficult about calculator programming; it's just learning a new (and quite simple) language and communicating in it. If you've managed to communicate with your drive-up bank teller machine, you'll find communicating with your HP-27S a breeze.

They're just words.

Getting Started

The Absolutely Worst Things That Could Happen... and Coping with Them

There are no wrong keystroke combinations. Regardless of all of the beeping and scolding your 27 throws at you (it might even show the ominous sounding "MACHINE RESET" and clear everything from memory), there's no way to damage the calculator by pressing keys.

The HP–27 communicates with us with messages asking for confirmation before deleting information or simply reporting mathematical errors. Instead of curtly not doing what we've asked (or worse yet, doing the wrong thing), the calculator tells us why it can't do it. These messages don't mean that you are using the calculator incorrectly, they are just a communications aid. If you don't like the beep, in a few pages we'll show you how to turn it off.

The most insidious calculator error is the one that goes by unnoticed. The 27 takes everything pretty much on face value; If our keystrokes don't cause some sort of mathematical or logical error, the calculator will return an answer. We'll try to help you recognize and correct this kind of error.

Resetting the HP-27S

No operations described in this book will cause a crash (freeze the display, or keyboard or show *that* message) if the directions are followed exactly. In the unlikely event of a crash (the keyboard locks up or the calculator acts strangely), you can reset your 27S by pressing and holding the [CLR] key while pressing the third [^] key from the left. Remember that this will clear the contents of memory. In fact the author has never seen a 27 crash.

Continuous Memory and the 10-Minute Time-out

The HP–27S is an eminently patient machine, though it's prone to catnaps. While it's waiting for keystrokes, the calculator wakes up from its nap every now and again (well, in calculator time, actually it's every 1/32 second) to refresh the display, then nods off again. In fact, given ten minutes of waiting for us to press a key, the calculator will time–out and shut down completely. You can press [■] [OFF] to turn off the calculator yourself in the middle of *anything*.

When you later wake up your calculator (by pressing [ON]), you can continue from *exactly* where you left off; the 27S doesn't forget a thing.

This isn't to say it's lazy. On the contrary, the 27 is one of the fastest pocket calculators available, but it was designed to squeeze about a year of normal use from a set of batteries.

Setting Display Contrast

It's usually easier to change the way the 27 display works than to change lighting conditions. You may find yourself changing the display contrast throughout the day to optimize as you move from desk to car to work station. Press and hold [CLR] while you press [+] to increase contrast (darken the display) or [−] to decrease contrast. Release the two keys simultaneously when the darkness looks right.

Keystrokes	Notes
[CLR] [−]	Press and hold down [CLR] while you press [−] to decrease contrast (lighten the display).
[CLR] [+]	Press and hold down [CLR] while you press [+] to increase contrast (darken the display).

The Keyboard

The 27S keyboard is designed to be used with one hand; with as many tasks as the calculator can handle, each key performs double (or triple or more) duties.

The Blue Shift Key [■]

Like a shift key on a typewriter, when you press the blue [■] key on the HP–27S, keystrokes take on the blue meanings printed on the face of the keyboard. You don't have to hold down the shift key; in fact, the calculator doesn't like it one bit, and will beep at you if you press another key while pressing [■]. A typewriter requires two hands (well, at least two fingers), while the 27 keyboard, because of this shift key, works fine with a single finger.

The little bent arrow symbol (↱) at the top left of the calculator display lights when you press [■], as a reminder that the next key you press will return its shifted meaning. For example, press [TAN] and you get tangent, but press [■] first and it becomes arctangent, or [■][ATAN].

If you change your mind, press [■] again to "un-shift" the keyboard.

Viewing Available Memory: The [■][MEM] Key

Creating statistical lists and formulas in Solver consumes memory. When you erase these lists and formulas, that memory is again available. The 27 can handle an amazing amount of information, though even with 6700 bytes available, it pays to be mindful of memory use.

This example shows all but 37 percent of calculator memory already being used by formulas, variables, and lists. If [■][MEM] shows a much smaller percentage, say under 350 bytes or about 5 percent, it's time to look for unneeded formulas and lists to delete. If you have a high percentage of memory available unused, you're not taking advantage of the power of the 27; use the memory, that's what it's there for!

Keystrokes	Display	Remark
[■][MEM]	AVAILABLE MEMORY: 2,614 BYTES 37%	Press and hold [■][MEM] for as long as you would like to view this message.

Erasing Data: The [■][CLEAR DATA] Key

Reclaiming memory used by Stat lists or Solver is easy, though remember: When it's gone, it's gone forever. In general, [■][CLEAR DATA] clears variables in the current menu, and a key called ≡DELET≡ in several menus erases data items. [■][CLEAR DATA] *always* clears the history stack and calculator line, though its other tasks aren't consistent in all applications; note the differences in this list:

Menu	Reaction
Main	Resets the history stack and places "0" on the calculator line. The calculator does not first ask for permission.
Solver	Deletes variables. The 27 first asks you if you'd like to delete the variables used in formulas. If you respond with ≣NO≣, nothing will happen. If you press ≣YES≣, the value from *all* Solver variables is lost and next prompt is: "DELETE ALL EQUATIONS?" CAUTION: pressing ≣YES≣ at this prompt will delete *everything* from Solver. Use ≣DELET≣ from the Solver main menu to delete a single formula.
Stat	Asks if you'd like to clear the *entire* current list. If you press ≣YES≣, the calculator will then ask if you'd like to erase the list name. To remove a single item from the active list, go to the main Stat menu and press ≣DELET≣.
TVM	Clears both the history stack and the TVM menu variables. CAUTION: the calculator does not ask for confirmation first.
%CHG	Clears *both* the history stack and the three %CHG menu variables.

The Difference Between [INPUT] and [=]

If you're familiar with HP RPN calculators, there's some comfort to be found in that big [INPUT] key. And if you are adept at algebraics, the [=] will make you feel right at home. Most often the 27 doesn't care which key you choose. At other times the difference is crucial. How do you decide which key to use?

Unlike [COS], [÷] or even [=], [INPUT] is not a function, but a terminator key. Pressing [INPUT] *always* says to the calculator that you've entered a complete operation, and now it's time to complete the task at hand; that's why it's called a terminator key. [=], on the other hand, often times be–comes a typing aid, simply entering the equals sign character.

In the [■][STAT] menu, [INPUT] enters the result of the equation into the current list, while [=] terminates the calculation *without* entering the result into the list. That's worth remembering, because you can still do keyboard calculations without changing data in the list, then enter a result in the list only if you need it.

If you're doing keyboard calculations, press whichever key is most convenient. Solver, the text editor, and several other operations require [INPUT], and as you become more familiar with your 27, you'll probably rely more and more on [INPUT] and less on [=]. That's why the [INPUT] key is so large.

View Full Numeric Precision: The [■][SHOW] Key

Early on we endorsed using ALL number display format because it shows the full precision of numbers all of the time. However, one discipline will prefer FIX 2, while another depends on SCI 3. If yours is one of those fields, you can still view the full precision of a number without wading through the [■][MODES] menu. Place the value on the calculator line and press [■][SHOW], and the value will remain in the display as long as you hold down the key. When you release the key the original display returns; nothing is changed.

Keystrokes	Display	Remark
2 [√x]	1.41	The square root of 2. This example assumes that the number format is fix 2.
[■][SHOW]	FULL PRECISION IS: 1.141421356237	Press [■] then press and hold [SHOW]; this display will remain as long as the key is pressed.

Notes about the Keyboard

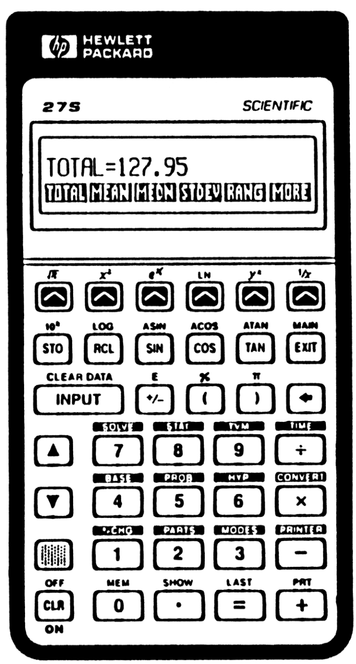
- You don't have to hold down the blue shift key [■]; press it once and release it. The annunciator at the top of the display lights to show that the shift key has been pressed. Turn off the shift key by pressing [■] a second time.
- You can use [INPUT] and [=] keys interchangeably during calculations. [=] becomes a typing aid when using Solver or the text editor.

- Press [shift] [MEM] to view currently available memory. The prompt will remain in the display for as long as you press the key.
- If you're using a FIX setting (or SCI or ENG...), pressing and holding [shift] [SHOW] will show the full numeric precision of the number in the calculator line.
- Error messages disappear when you press the next key. You can also erase the error message without affecting anything else by pressing [←] or [CLR].

Navigating the Menus

As calculators have grown more powerful, the problems in using these new capabilities have compounded. With a key for each function, we'd end up with a keyboard the size of a cookie sheet. Shift keys are a more practical solution: shifted [•] becomes [shift] [SHOW]. Again, as capabilities increase, so does the number of shift keys. A more workable solution is sub–menus, with groups of related functions. The [shift] [TIME] menu houses all time–related functions, and the calculator still fits in your pocket.

Living beneath the number entry and most function (÷, x, −) keys are the keys (both literally and figuratively) to the power of the 27S.



Menu Keys: [⌃]

Press [■], the shift key, followed by one of these menu keys and the top row of [⌃] keys, called menu keys, are dedicated to a special task, and the bottom line of the display shows the meaning of these keys.

If even one of these keys is used in a menu, all of the top–row keys be–come menu keys. You can still use the functions assigned to those keys by pressing [■] first.

So, once you're in the menu, how do you get out, and more importantly, how do you go to other menus? With a few restrictions that we'll discuss in a few minutes, to jump to another menu, simply press the menu key. Two special keys handling traversing between menus:

Keystrokes	Reaction
[EXIT]	Back up *one* level in the menu structure.
[■][MAIN]	Exit from the menu completely and return to the main calculator display.

Menus and Modes

If you want a comma instead of a decimal point, Radians instead of Degrees trig mode, or numbers rounded to two decimal places, you can have it. The 27S is customizeable; you can run it how you want to, not how some engineer in far off Corvallis deemed you should.

The [■][MODES] and [■][PRINTER] menus are the gateway to customizing the calculator. The [■][MODES] menu sets number display format, Degrees/Radians mode, printer AC adapter/battery toggle, and status of the beeper. To get a feel for how menus work, let's walk through the pro–cedure for turning off the beeper.

Turning Off the Beeper

In this example, the calculator line contains the value 17.4, and the second line contains zero. First we'll press [■][MAIN] to make sure that the calcu–lator is at the main display. At the main display, no menus are shown and we do calculations on the bottom line.

Keystrokes	Display	Remark
■ MAIN	0 17.4	This is the main menu display; no menus are displayed. Both lines show numbers.
■ MODES	SELECT DISPLAY FORMAT FIX | SCI | ENG | ALL | ./, | MORE	The Modes menu.
MORE BEEP	BEEPER ON: APPTS ONLY D/R | BEEP | PRNT | | | MORE	Move to the second submenu then press BEEP to toggle the beeper on or off.

Press BEEP repeatedly, cycling through the three options of "BEEPER ON," "BEEPER OFF," and "BEEPER:APPTS ONLY." Whichever is the last in the display remains the active mode. "BEEPER:APPTS ONLY" is usually preferred to "BEEPER OFF" because alarms will sound, while errors and misplaced keystrokes will go unnoticed by co-workers.

In addition to the "BEEPER OFF" message, you can test if the beeper is on or off by pressing one of the un-labeled (⌃) keys.

When you're done, press EXIT to return to whatever you were doing before the exercise. Since we were at the main display, it will return:

Keystrokes	Display	Remark
EXIT	0 17.4	When we press EXIT, the main display returns. The value in the calculator line is not changed.

Number Display Format

Unless otherwise noted, we'll print numbers in this book in ALL number format. ALL format shows only as many decimal places as are necessary to represent the full precision of a number. If the number is an integer (a whole number without a fractional part), ALL shows neither trailing zeros nor a decimal place. The only time ALL format may be a hindrance is showing very large or very small numbers, when instead of showing 0.123456789012, the calculator shows the value in scientific notation as 1.23456789012E−1.

Regardless of how the values are shown, the calculator always stores and calculates with the full precision of 12−digit mantissa and 3−digit exponent.

If you are more comfortable with FIX, scientific or engineering notation, feel free to change. Press [■][MODES] to enter the MODES menu, then ≡FIX≡, ≡SCI≡, or ≡ENG≡ followed by a value of zero through eleven (for the number of decimal places), then press [INPUT]. ≡ALL≡ returns the number format to suppressing unnecessary trailing zeros.

Keystrokes	Display	Remark
[■][MODES]	SELECT DISPLAY FORMAT FIX | SCI | ENG | ALL | ./, | MORE	Enter the MODES menu then press the appropriate [△] key.
≡FIX≡	TYPE #DIGITS (0-11); PRESS [INPUT]	This example shows the ≡FIX≡ key, but ≡SCI≡, ≡ENG≡, or ≡ALL≡ are other available formats.
[4] [INPUT]	4▒ PRESS [INPUT]	Now press number keys for the FIX setting, in this case [4], then press [INPUT].

After these keystrokes the 27 is in FIX 4, showing four decimal places. In the same menu, press ≡ALL≡ to return to the default number format.

Digit Separator

Also in the [■][MODES] ≡MORE≡ sub−menu, ≡./,≡ toggles between showing numbers with a period or comma separating digits. If you'd prefer the number one and a half displayed in the European format of "1,5" instead of "1.5", press [■][MODES] ≡./,≡. Like FIX, the digit separator has no effect on how the 27 handles numbers internally, just how it displays them; you can toggle between number formats without affecting calculations.

Angular Mode

We'll show examples in Degrees trigonometric mode, with all angles measured in decimal degrees, rather than in Radians mode which calculates using degrees, minutes and seconds. From the [▒][MODES] menu, select ≡MORE≡ for the second screen of options.

Keystrokes	Display	Remark
[▒][MODES]	SELECT DISPLAY FORMAT FIX | SCI | ENG | ALL | ./, | MORE	Enter the [▒][MODES] menu.
≡MORE≡ ≡D/R≡	SELECT MODE D/R | BEEP | PRNT | | | MORE	At this second menu, press ≡D/R≡ to toggle between Degrees and Radians trigonometric modes.

An annunciator at the top of the display showing "RAD" lights when the calculator is in Radians mode; in Degrees mode (the default) the annunciator isn't displayed. When you've chosen the angular setting you want, press [EXIT] to return to the previous menu, or [▒][MAIN] to return to the main menu (with no menu keys displayed).

Notes about Menus

- Changing menus during a calculation acts as a number terminator. The reason for this is simple: if you change the number format, the appearance of numbers in the formula in the calculator line will change.
- When you're in a menu, the top row of keys is "taken over" by the menu. You can still use the key's original function by pressing [▒], the shift key, first.
- Pressing [EXIT] steps back a single menu level.
- [▒][MAIN] always returns the calculator to the main menu display.
- ≡MORE≡, the rightmost [△] key in several menus, takes you to a second set of key definitions. To return to the original menu, press ≡MORE≡ repeatedly until it "comes around again."

Introduction to Calculations

If You're Familiar With HP Calculators... Or Maybe You're Not

Pick up a traditional HP calculator and the first thing you know, it's beeping and flashing a flurry of error messages. Regardless of your experience with calculators, your 27S began earning it's keep, right out of the box. So how is the HP–27S different from other HP calculators?

Hewlett–Packard has built a reputation from powerful, reliable, compact calculators; but mostly they've been RPN (Reverse Polish Notation, or post–fix math) calculators. As you likely realized right away, the HP–27S isn't an RPN calculator. If you're not a long–time RPN calculator user (or maybe you are, but you also need the services of an algebraic calculator), this is a moot point.

With an RPN calculator we can more readily control the values on the stack (in fact, we have to) and the order in which the operations are evaluated. An RPN calculator requires a constant awareness of the status of the stack, and demands both numbers on the stack before we perform an operation. That's why several million people use Hewlett–Packard RPN calculators. And that's probably also why many millions more use algebraic calculators and why the HP–27S is an algebraic calculator.

RPN AND ALGEBRAIC LOGIC	
Keystrokes	Calculator
2 ENTER 3 *	Hewlett–Packard RPN Calculator.
[2] [x] [3] [INPUT]	HP–27 True Algebraic Calculator.

During calculations, the algebraic calculator remembers the first number when we press [x], then enters the second number and performs the mathematical operation when we press [=]. There's a stack holding intermediate values, but the calculator manipulates it automatically.

Each of these approaches has one thing in common: A single keystroke tells the calculator that you want it to process the formula and return an answer. With the RPN calculator, it's the star (*) key; computers often use the star (or asterisk) as the multiplication symbol.

Algebraic logic places the operator (+, −, ÷, etc.) between the arguments (numbers), and the equals [=] key (or [INPUT], you have your choice) completes the expression. You calculate as you think: "Let's see... two times three equals six." Algebraic logic eases the translation from blackboard logic to keystrokes.

The Calculator Line

The HP–27's calculator line is the working line in the display; it's where the current calculation unfolds. When a menu is active, the calculator line moves to the top of the display, then when you exit the menu the calculator line moves back to the bottom of the display. And sometimes, such as when we work in Solver or store a number in a register, the calculator line isn't displayed at all.

Let's work through a simple calculation. Remember to press [■][MAIN] before starting if there's a menu at the bottom of the calculator display. Notice that the cursor flashes while you edit a number or formula.

Keystrokes	Display	Remark
2 [√x]	4.5678 1.41421356237	The calculator line is the bottom line on the display. Press [√x] for the square root of 2.
[■][PARTS] ≣FP≣	0.41421356237 IP | FP | RND | ABS	The calculator line moves to the top of the display when we enter the [■][PARTS] menu.

[▒][MAIN]

```
4.5678

0.41421356237
```

We no longer need the [▒][PARTS] menu so we exit; the calculator line moves back down.

What Did We Just Do? More about [INPUT].

We moved from the main display (with no menu keys on the bottom line) through the [▒][PARTS] menu and back to the main display as we worked through this calculation. When you need a function not in the current menu, just pop into the needed menu, poke the appropriate [△] key and return, all without losing your place in the formula.

A second point is that we didn't press [=] or [INPUT] at all. It wasn't nec–essary, since [√x] completed an expression. Then ≡FP≡ marked the end of the next calculation. This is a chain calculation, a finished product made of several sub–assemblies.

Somehow we have to tell the calculator that we've entered a complete number. Otherwise it'll happily wait, cursor flashing, forever. That's easy: Pressing a function key marks the end of a number. And that's what you'd do anyway, so you don't even have to think about it.

If you are at the end of the calculation, obviously you aren't going to press another function key (it would trash that hard–earned number on the calc–ulator line), so you press [INPUT] or [=]. A true algebraic calculator calcu–lates formulas how you would say them; you rarely say "equals" in the middle of a calculation.

Entering Parentheses

We'll discuss parentheses in detail in few pages, but since this section shows how simple and versatile calculating with the 27 is, we should show how, well, simple and versatile parentheses are. With parentheses, you can control how a formula is evaluated and, unless you really want to use one, you won't end up with an incredibly long formula winding it's way through the hidden belly of the calculator.

This example works through the formula 2x(3+4)–(5x6). You would say it as "Two times the result of three plus four, minus the result of five times six." And that's how we'll key it in:

Keystrokes	Display	Remark
2 [x] [(] 3 [+] 4	2x(3+4▒	The 27 calculates formulas in parentheses first. Parentheses have priority over multiplication.
[)]	2x7	The expression within parentheses is calculated. The HP–27 waits for the next keystroke.
[−]	14-▒	The calculator evaluates the multiplication. Cursor is flashing, the calculator expects a number.
[(] 5 [x] 6	14-(5x6▒	Enter the expression in the last set of parentheses. Subtraction is outside of the parentheses.
[INPUT]	-16	Press [INPUT] to complete the calculation. The 27 First evaluates the formula in parentheses, then sub– traction.

We didn't enter a closing parenthesis, but instead pressed [INPUT] to mark the end of the expression. This saved a keystroke, but that's only part of the work the 27 does automatically for us. In fact if you have a complex formula with several opening parentheses, pressing [INPUT] evaluates the formula and adds closing parentheses to all of them.

Had we entered the closing parenthesis, the partially evaluated formula would remain in the display. We would still have to press [INPUT] before the calculator would completely evaluate the formula.

```
14-30
```

If we had pressed [)] instead of [INPUT] at the end of the example above.

Adding a final closing parenthesis completes an expression; with no ambiguity about what's to be done next, the 27 simply evaluates the expression. In fact, adding a final closing parenthesis will evaluate a formula... even if the formula didn't contain parentheses at all, something like a spare [=].

Erasing the Calculator Line: The [←] and [CLR] Keys

In typical HP–27S style, there are a couple of ways to erase the number on the command line. In fact, if you're used to conventional calculators, you probably pressed [CLR] the first time to turn off the calculator, and end up with the number zero in the calculator line. That's one way to clear the number. For most purposes, [←] is a bit more elegant.

When the cursor isn't on and you aren't in the middle of a formula, [←] works like the [CLR] key, resetting the calculator line to zero. However, during calculations, [←] "backs you out" of a formula, deleting the right-most number or operator. Instead of starting over, you can erase just the error, and continue from that point.

The History Stack

Now you might be saying "What's this *stack* business, I thought I bought an algebraic calculator?" Like a printed tape, the 27S records the answers from previous calculations. Though only two lines are displayed, the calculator remembers the results of the last four calculations. And unlike a printed tape, you can reuse these numbers in later calculations. You can roll the tape back into *this* adding machine and use the numbers again!

If the purpose of a stack is seems bit vague and unnecessary, walk through these examples. The 27 would still be a powerful calculator without the history stack, but once you're comfortable with the stack, you likely won't want to go back to a machine without one.

Keystrokes	Display	Remark
2.5 [INPUT] 1 [INPUT]	2.5 1	This enters a couple of numbers in the history stack.

If you could see all four entries, the history stack would look something like:

```
4.5678
0.41421356237
.............................
2.5
1
```

Moving an entry from anywhere in the history stack to the calculator line is just a matter of pressing [▼] to rotate down, or [▲] to rotate up through the stack. It's circular; one number's ceiling is the next number's floor.

Keystrokes	Display	Remark
[▼]	1 4.5678 .... 0.41421356237 2.5	Press the [▼] key once and the stack rotates down. The stack will rotate up if you press [▲].

The next formula you enter moves the contents of the history stack up one level, pushing the oldest number (in this case the value 1) off of the top of the stack. In other words, a fifth calculation, when four values are already on the history stack, causes the *oldest* value to be lost. Where that fifth number goes is a mystery.

Keystrokes	Display	Remark
	4.5678 0.41421356237	
7 [÷] 4 [INPUT]	2.5 1.75	This latest calculation pushes the oldest number out the top of the stack.

Working With the History Stack

Remember that when we begin a new calculation, the old contents of the calculator line move up the history stack to make room for this new formula. This is often called "stack lift." And in the same vein, when we reuse the number in the calculator line in a new formula, the stack doesn't move at all.

There's yet another way to control the contents of the history stack: Press [CLR] or [←] *before* starting a calculation to erase the number on the calculator line. That way you can enter a new formula without changing the rest of the history stack.

If you just begin entering a formula, the contents of the calculator line will move up level in the history stack. The rules the calculator follows regarding stack lift are simple, in fact once you know these rules, you can forget them; they work just how you'd expect them to:

- To enter a formula and preserve the result from the previous formula in the history stack, simply enter the new formula.
- To write a new formula without changing the contents of the history stack, press [CLR] or [←] to reset the calculator line before entering a formula.
- To use the value from the calculator line in a new chain calculation, begin by pressing an operator key (such as [+]).
- If the contents of the history stack are not important, simply enter your formula and don't pay any attention to the stack. It's not even necessary to press [CLR].

While the history stack isn't the place to store important data (the 27 has registers for that, and we'll talk about them in a few minutes), you can keep several values floating on the stack, and reuse numbers in chain calculations.

History Stack Keys

Keystrokes	Reaction
[▲]	Rotates the history stack up one item.
[▼]	Rotates the history stack down one item.
[CLEAR DATA]	Erases the contents of the history stack and leaves the value zero in the calculator line.
[■][LAST]	Recalls the value from the second level of the history stack to the current formula.

LAST: The [■][LAST] Key

As we've seen, if you want to build from a previous result, you can rotate the stack until the value you need is on the calculator line. However, what do you do if you want to use that value within a formula, or maybe even use it several times in a calculation?

Pressing [■][LAST] while you enter a formula inserts the second number in the history stack (the top line in the display) in your formula. This is quite helpful when you need to see an intermediate result, then use that value again in ensuing calculations. Since the stack is a movable thing, whichever value is in the second line is considered the *last* result, though as we've been experimenting, that's not always the case. Press [▲] or [▼] to rotate the stack, and any one of those four values can qualify as *last*; the only requirement is that it be the second item on the history stack.

The [■][LAST] function is probably the least intuitively named key on the calculator, and the reason is simple: Hewlett–Packard RPN calculators have a key called "LastX" which really does recall the value from the previous calculation. In the case of the 27, that's not necessarily what it does. But enough grousing, using [■][LAST] *is* intuitive, even if the name isn't.

Keystrokes	Display	Remark
2 [+] 3 [x] 4 [−] 5 [INPUT]	9	Enter a formula and press [INPUT]. Entering the next formula moves this value up on the stack.

2 [x] [■] [LAST]	9 2x9▒	[■] [LAST] copies the value from the previous calculation into this formula.
[INPUT]	9 18	Now the results of both calculations are on the history stack.

When the Calculator Calculates a Formula

Early on we called [INPUT] a terminator key because it marks the end of a task. But other keystrokes communicate to the 27S that we want to start a new calculation or continue from previous result on the command line. The rules to follow when deciding how to begin a calculation are simple:

- If you want to use the number from the calculator line in a new calculation, simply press an operator key and continue entering the formula.
- If you don't want to use the number from the calculator line, simply ignore it and begin typing a new formula. You can press [CLR] or [←] to erase the calculator line before entering the formula, but it's not necessary. We'll talk more about [CLR] and [←] later in this chapter when we cover the history stack.

From the calculator's point of view:

- When you begin by typing a number, the calculator assumes that you are beginning a new formula.
- When you begin by typing a function key, the calculator assumes that you want to continue working with the number in the calculator line.

As we work through a formula the calculator is always on guard, monitoring our keystrokes and ready to evaluate intermediate results and reduce the formula to the simplest expression in the calculator line. We've seen this work in the examples above, but as formulas get more complex this power of the 27 becomes more evident.

From the calculator's point of view, it's a lot of work, but that's what it does for a living. From our point of view, the more work the 27 does for us the better.

Operator Priority

We're going to complicate things for a few minutes in order to make things simpler to understand later. If you've entered formulas and found the results weren't exactly what you expected, it's probably because calculators don't calculate like people do; they unerringly follow rules and do everything consistently.

Instead of interpreting formulas from left to right, the 27S follows rules of operator priority, also known as mathematical precedence. As the calculator monitors our keystrokes and simplifies formulas, these are its guidelines. Operator priority, simply put, is the order in which mathematical expressions are evaluated. To avoid having to scatter voluminous parentheses in everything we do, and to avoid occasional surprises on the bottom line, it pays to be conversant in operator priority.

OPERATOR PRIORITY	
( )	Expressions in parentheses.
ABS RADIUS SIN ...	Functions.
y^x	Exponentiation (^).
x ÷	Multiplication and division.
+ −	Addition and subtraction.

In other words, when we enter formulas without parentheses, the 27 evaluates functions following the rules in the table above. Multiplication, for example, gets calculated before addition, regardless of which we enter first. That also explains why some parts of formulas seem to hang around on the calculator line forever, and others disappear before we lift our finger from the key. Operations of the same level are interpreted from left to right. That's true algebraic, as opposed to the garden variety calculator.

We've been using parentheses and following operator priority in compound expressions all along, but let's look at another (2+3x4−5), entered without parentheses, and watch how the calculator handles it. Notice how this sim–

ple formula gives different results as we work through the next few examples. First we'll walk through the formula without parentheses:

Keystrokes	Display	Remark
2 [+] 3 [x]	2+3x▒	Addition follows multiplication so the addition calculation waits to be calculated.
4 [−]	14−▒	Multiplication is completed, then the addition.
6 [INPUT]	8	The expression is completed.

Note again, multiplication is evaluated *before* addition, so the intermediate result is based first on the multiplication, then the addition. Since addition and subtraction are on the same level of precedence, the 27 interprets from left to right, as they appear in the expression. The 2+3 hangs there while the calculator waits for us to enter the next expression.

Parentheses

Operator priority usually works to our advantage, saving time and simplifying keystrokes. But not always.

When a calculation becomes unwieldy, we can enclose expressions within parentheses to insure that the calculator evaluates them as intended. Expressions contained in parentheses are evaluated together, regardless of what functions live beyond their borders.

If you're not sure of how the calculator will evaluate an expression, add extra parentheses for clarity; leftovers are ignored. The keys we press are shown on the left, intermediate results in the middle.

Keystrokes	Display	Remark
(2 + 3)	5	The closing parenthesis completes the operation.
x 4 −	20-▒	Multiplication before division.
5 INPUT	15	The expression is complete.

Moving the parentheses around, we can come up with various results from the same formula:

Keystrokes	Display	Remark
(2 + 3) x 4 − 5 INPUT	15	The formula: (2+3)x4−5
2 + 3 x (4 − 5) INPUT	-1	The formula: 2+3x(4−5)
(2 + 3) x (4 − 5)	-5	The formula: (2+3)x(4−5)

Entering a closing parenthesis, even though one is already displayed, will evaluate the expression as far as possible, and move the cursor outside of that set. The 27 will automatically enter a closing right parenthesis for each left parenthesis entered when you press INPUT, though Solver requires matching sets.

Notes about Operator Priority and Parentheses

- The 27 is a true algebraic calculator; it performs each operation in chain calculations following operator priority rules (which type of function the calculator performs first), and shows intermediate results as we work through a calculation.
- After making such a big deal at the outset, we've rarely used INPUT in these examples. Like shutting off the lights at the office the just before you go home, pressing INPUT is not something you plan your calculating day around.

Notes about Calculations

- If you press a function key, the calculator knows to perform the function on the number on the calculator line. However, if you begin the expression by entering a number (or parenthesis), the calculator realizes that you are beginning a new formula.
- The HP–27S follows the mathematical rules of precedence (operator priority). If you aren't sure of a rule, or just want to change the order for a formula, enclose parts of the formula within parentheses.
- The calculator ignores extra sets of parentheses, and they're free; when in doubt use several.
- Some algebraic calculators and computers (including HP BASIC) perform exponentiation *before* functions.
- Some earlier HP algebraic calculators ignore operator priority. The HP–18C, for example, evaluates expressions from left to right.
- Either [INPUT] or [=] will terminate a formula.
- If a formula grows too long to show on one line, the 27 shows a set of ellipses (...) and as much of the formula as will fit.
- It isn't necessary to enter closing parentheses, the calculator adds these right parentheses when you press [INPUT] as the last step in a calculation.
- You can change menus during calculations, however if you change menus while entering a number, the menu key acts as a number terminator.
- While accounting practices often place negative numbers in parentheses, for example (1) to represent minus one, the 27 precedes negative numbers with a minus sign: –1.

CHAPTER FOUR

Registers

As clever as the history stack is, it's at best transitory. HP has given us ten calculator registers (numbered zero through nine) for more secure storage. While just about every calculator has registers of one kind or another, please work through this section because HP has added some new twists.

[STO] and [RCL] *copy* the value from one place to another; specifically, between the calculator line and storage registers.

These examples copy values from the calculator line to registers 3 and 4. We'll do some calculations with the numbers, but that's just to keep the numbers interesting; [STO] and [RCL] are the main point of this exercise.

Keystrokes	Display	Remark
2.65	`2.65`	Enter a number (2.65) in the calculator line.
[STO]	`STO _`	The calculator is asking you to enter a register number. We'll press [3] for register number three.
[3]	`2.65`	The number from the calculator line is copied to register three. Finally the calculator line returns.

[+/-]	-2.65	Change the sign of the number on the calculator line.
[STO] [4]	-2.65	Now register 4 contains −2.65. The number in the calculator line returns.

Notice that after each [STO], the value in the calculator line remains unchanged; [STO] *copies* the value to the register. Recalling a value from a register is much the same:

Keystrokes	Display	Remark
[RCL]	RCL _	The calculator is asking for a register number. We'll press [3] for register number three.
[3]	-2.65 2.65	Contents of register 3 are copied to the calculator line. Previous calculator line moves up the stack.

Recalling the value from a register *does not erase* the register; the value will still be there the next time you need it. If you followed the above keystroke examples, register 3 contains the value 2.65 and register 4 contains −2.65, and will until you store something else in them.

Both [STO] and [RCL] *copy* the value from one location to another.

Registers in Chain Calculations

PI is a constant, it never changes; pressing [■][π] (PI) returns 3.14159265359 *every* time. A constant is a value, like PI, you often use. Every time you store a value in a register, you create an impromptu constant. The only reason to store a value in a register is if you intend to use that value again.

However, unlike [■] [π], there's no label showing the contents of registers; what if you've forgotten which register has the value you need? The 27 lets you "try them on for size" until you find the one you want. Press [RCL] then a register number until you uncover the number.

Keystrokes	Display	Remark
7 [x] [RCL] [3]	`7x2.65`	Begin a formula; recall the value 2.65 from register three.
[RCL] [4]	`7x-2.65`	The contents of register four (−2.65) replace the last number in the formula.
[INPUT]	`-18.55`	Complete the calculation.

If, after all of that, you find that none of the registers seem to fit, simply type a new number, it'll replace the last number.

Storing in a Register During Chain Calculations

Pressing [STO] in the middle of a chain calculation copies the *rightmost* number in the calculation to the register. If there's only one number in the calculation, [STO] uses that number.

Keystrokes	Display	Remark
4 [÷] 5 [+] 6	`-18.55` `0.8+6▒`	An intermediate result, then continue the calculation.
[STO] [2]	`-18.55` `0.8+6`	Store the value 6 in register number two. Notice that the stack doesn't change.

[−]

```
-18.55
6.8-▒
```

Continue the chain calculation.

[RCL][2]

```
-18.55
6.8-6
```

Prove that register two contains the value 6.

[INPUT]

```
-18.55
0.8
```

The chain calculation is completed.

Storage Register Arithmetic

Normally, storing a value in a register overwrites the previous contents of that register. However, what if you want to add the value from the calculator line to the contents of a register, as a running total?

The 27 makes short work of storage register arithmetic: Simply press [STO][+][0] to add the calculator line to the current contents of register zero. It works the same with [÷], [x], [−] and [y^x] (exponentiation) and any of the ten registers.

The value on the calculator line is unchanged by storage register arithmetic; the function is performed on the number in the storage register. We'll use register zero in this table, but storage register math works with registers zero through nine:

Keystrokes	Reaction
[STO][+][0]	Adds the calculator line to register zero. That is, «register» = «register» + «CalculatorLine».
[STO][−][0]	Subtracts the calculator line from register zero. That is, «register» = «register» − «CalculatorLine».
[STO][x][0]	Multiply the calculator line and register zero. That is, «register» = «register» x «CalculatorLine».
[STO][÷][0]	Divide register zero by the calculator line. That is, «register» = «register» ÷ «CalculatorLine».

[STO] [■] [y^x] [0]
Exponentiation. Raise register zero to the power of the calculator line. That is, «register» = «register»^«CalculatorLine».

Storage register math can be difficult to visualize because, well, there's nothing to see; everything happens to an unseen register.

Keystrokes	Display	Remark
5 [STO] [0]	0.8 5	Register zero contains the value 5.
[STO] [x] [0]	0.8 5	Multiply the number in the calculator line (5) with register zero. Calculator line is not changed.
[RCL] [0]	5 25	To prove that register zero contains the result of 5x5. The stack lifted.

Keep in mind the order of operators when you're working with division and exponentiation. In algebraic fashion, the register is on the left and the value of the calculator line is on the right of the operator.

Register zero still contains 25; let's look at storage register exponentiation.

Keystrokes	Display	Remark
2	25 2▒	Enter 2 in the command line. Remember that register zero contained 25 before we started.
[STO] [■] [y^x]	25 STO ^ _	The calculator is asking you to name the register on which to perform exponentiation.

Keys	Display	Description
[0]	25 2	Press [0] to specify register zero. The calculator line returns.
[RCL] [0]	2 625	Prove that register zero contains the result of the formula 2^25.
[STO] [▒] [y^x] [0]	2 OVERFLOW	An overflow error because 625^625 is beyond the range of the calculator.
[RCL] [0]	25 9.99999999999E499	Since the calculation is in a register, the overflow occurs there.

Remember, when an overflow or underflow error occurs, the calculator returns the largest or smallest available number to the calculator line. Since this is a storage register calculation, the overflow occurs in the register, and the value is returned to the register. The error message is displayed, but the calculator line is not changed.

Clearing Calculator Registers

If you've been poking around looking for the equivalent of a [▒] [CLEAR VARS] key for the ten calculator registers, you're in for quite a search. There's no memory to be gained; the registers are there regardless of if there's anything stored in them or, for that matter, if there are no batteries in the calculator (that's a sure way to clear the registers).

There's usually little to be gained from clearing registers with any calculator, but without a way to do it there's always a catch. When a register contains zero, it's likely that it's unused, and that's the logic in clearing them. However, zero is often a valid result of calculations (statistically it's probably a more likely result than most others).

Deciding if a register contains a given value because it is unused, or the misbegotten remainder from another days work, can be quite a quandary. The moral: Keep track of what's where.

Pressing [■] [CLEAR DATA] clears the calculator *history stack*, not the registers. If you really want to clear a register, store zero in it:

0 [STO] [1]

Notes about Calculator Registers

- Calculator registers work with the value in the calculator line. If you want to store another item from the history stack, press [▲] or [▼] to rotate up or down until the item you want is in the calculator line.
- The "STO_" or "RCL_" prompt covers up the calculator line. The calculator line does not move up as it does when you enter a menu (it may disappear for a second). The line will return when you've completed the operation.
- Recalling and storing values do not terminate formulas. You can incorporate register values into formulas.
- There is no command to clear variables. If you want to reset a variable to zero, simply store zero in it.
- Memory for registers does not come from calculator memory; they always exist, even if you don't use them. It doesn't cost any extra memory to use the calculator variables, so use them freely.
- None of the menus use calculator registers for their own purposes; they are truly your personal storage.

Putting it All Together

Powerful calculator isn't an oxymoron, like "jumbo shrimp" or "readable user's manual." Each new calculator is the sum of experiences learned from earlier machines. And since Hewlett–Packard engineers have been creating handheld scientific calculators since 1972, by the time they got to the HP–27S, this library of functions became extensive.

A Library of Functions

Using these functions is consistent and nearly self–explanatory, so we'll show just the [■] [PARTS] menu, which is representative. The reference section in the back of this book includes examples and a few restrictions for using the functions in these menus. This is a list of the keys and menus associated with the numeric function library.

Keystrokes	Reaction
[EXIT]	Exit from the numeric function menu.
[■] [BASE]	Enter the Decimal/Hexadecimal/Octal/Binary base conversions menu.
[■] [HYP]	Hyperbolic and inverse hyperbolic functions menu.
[■] [PARTS]	Extract a part of a number.
[■] [PROB]	Probability menu; combinations, permutations, factorials, and random numbers.

Just by looking at the 27's keyboard, you know that we've left out more menus than we've included in this list. That's because many of these other menus are *applications*, or pre–written formulas, rather than collections of *functions*. This isn't an arbitrary distinction (which isn't to say that we won't be making arbitrary distinctions while discussing them), but a clari–

fication of how menu keys work in various menus. We'll explain these distinctions when we get back to the rest of the menus; remember, we haven't even mentioned Solver yet.

Parts of Numbers: The ■ PARTS Menu

Each of these menus contains groups of related functions. The goal being to make them easy to find, but unobtrusive when you don't need them. In the ■ PARTS numeric function menu, IP returns the integer portion, FP returns the fractional portion, RND returns the number rounded to the current fix setting, and finally, press ABS for the absolute value (magnitude) of a number.

Enter the function menu, select the menu key, and continue your calculation, moving from menu to menu as the calculation warrants.

Keystrokes	Display	Remark
2 √x ■ PARTS	1.41421356237 IP | FP | RND | ABS	Enter a number with a fractional part. Then, wherever you are, just press ■ PARTS.
FP	0.41421356237 IP | FP | RND | ABS	Get the fractional part of the number on the calculator line.

The operation is complete, but the ■ PARTS menu remains in the display. Press EXIT *once* and you'll return to from whence you came. Each numeric function menu works the same; easy in, easy out.

Notes about Numeric Function Menus

- You can change function menus during calculations.
- You can enter another numeric function menu directly from any other numeric function menu; it isn't necessary to press EXIT first.
- You can enter function menus from other menus; press EXIT *once* to return to the previous menu.
- When in Solver, function menu keys become typing aids.

- Notice that when you enter a menu and use a [△] key, you remain in that menu. Some HP calculators (like the 22S) exit the menu after you press the function key.

Base Conversions: The [■][BASE] Key

We'll leave our short tour of this function library with a look at base conversions, a relatively new application, and likely foreign to the average 27 fan.

People are decimal-oriented animals, having been so since the first human counted ten fingers, or at least for the last couple thousand years since somebody invented zero. With this heritage, computer science began with decimal numbers. Unfortunately, the computer's complete lack of fingers and inability to count above one soon forced the industry to use binary (called base-two) math. And that's where we are today; experiments in neural networks to correct this weakness are probably ten years from fruition. In the meantime, the [■][BASE] menu translates computer math to decimal. And the ≡BIN≡ key doesn't open a secret compartment in the calculator, but switches to binary mode.

Keystrokes	Display	Remark
[■][BASE] 2766 [+] 73	2,766+73▓ [DEC] HEX | OCT | BIN	We're in the [■][BASE] menu. The calculator is set to decimal mode.
≡BIN≡	101011001110+1001001 DEC | HEX | OCT | [BIN]	When we press ≡BIN≡, the calculator switches to binary display format. The numbers are unchanged.
≡HEX≡	ACE+49 A | B | C | D | E | F	Numbers are displayed in hexadecimal; the menu keys change to hex typing aids.

[EXIT]
[INPUT]

```
2839
[DEC] HEX  OCT  BIN
```

Press [EXIT] once and the BASE menu returns. The calculator is in decimal mode.

In each of these cases the number remains the same, just the displayed representation changes. Calculations are shown in the current base.

Decimal, Hexadecimal, and Octal are alternate ways to view and work with binary numbers. Binary digits or "bits" (a contraction of parts of both words) can have a value of either zero or one. This limited range is made usable by grouping units of four bits into a nibble or, as we'll call it throughout this book, a nib (an alternate spelling is "nybble"). Various combinations of zeros and ones in these four bits represent values from zero (all bits clear) through fifteen (all bits set). So, hex and Octal are convenient ways to work with binary numbers.

BASE CONVERSIONS			
Decimal	Hex	Octal	Binary
0	0	0	0000
1	1	1	0001
2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
10	A	12	1010
11	B	13	1011
12	C	14	1100
13	D	15	1101
14	E	16	1110
15	F	17	1111

Binary 10 (one zero) plus 1 is 11 (one one), but 11 plus 1 is 100. Follow down the right-hand binary column in this table and you'll see that every time you add one to a value whose least significant bit (the bit on the right) is one, the one is carried to the left. Binary numbers are right–justified, most often showing leading zeros.

In hex, "9" plus "1" is "A"; values between 10 and 15 are represented as a single digit of "A" through "F". Values larger than "F" (15 in decimal) are represent in groups of nibs called words. A common word size is two nibs, or one byte, though word size depends on the application. In general, word size is the same as CPU register size.

Octal, like hex, is a way to represent groups of bits, though carry occurs after 7: "7" plus "1" is "10".

You can't set the 27 to a limited word size; it's always 36 bits. Be cautious simulating large numbers which should set a carry bit to indicate overflow.

Notes about Base Conversions

- Word size is always 36 bits. The largest value in decimal is 34,359,738,367, in hex it's 7FFFFFFFF.
- The error message "TOO BIG" occurs if you try to convert a decimal value which is greater than 36 bits.
- Fractional parts of decimal numbers are ignored.
- Press [+/–] for two's compliment calculations.
- You can enter numbers in any of the four bases and switch to another base to do calculations.
- When you exit the base conversion menu, the number on the calculator line is converted to decimal.
- The leftmost bit is the sign of the number, it's zero (clear) for positive values, one (set) for negative values.

CHAPTER SIX

The Text Editor

This is an introduction to the text editor, also called the alpha menu. If you'd like see more about the text editor, even a way to use it as a mini-database, after reading this section, skip to the chapter called "Introducing Solver."

A Text Editor in a Calculator?

Why do you need a text editor in a calculator? An advanced calculator like the 27S does not live by numbers alone; the [■][TIME], [■][PRINTER], and [■][SOLVE] menus all use the text editor. So how do you crunch an entire typewriter keyboard into six menu keys? With a lot of keystrokes!

While you probably wouldn't want to write a long report with the 27S text editor, it'll handle the tasks of appointment memos just fine. And in Solver, the main reason for the text editor, it's surprisingly versatile.

The cursor (the blinking ▒ character) marks your position on the line. The text editor inserts each character you type at the cursor position and any characters from the cursor–on move to the right, much like word processor insert mode on desk–bound computers.

Type the characters you want, pressing [←] to erase any mistakes, or [CLR] to erase the entire line to start over. Then press [INPUT] (the terminator key) when you're through editing.

A line of text is limited to the 22 character window unless you're in Solver. When you reach the end of the line, the calculator will beep a warning.

If you don't press [INPUT], you can turn off the calculator and leave a note in the display for the next time you use the calculator. When you return to the calculator, simply press [CLR] to erase the line, or [■][MAIN] to leave things unchanged.

The [■][PRINTER] menu ≣MSG≣ window is a convenient place to leave a temporary message (even if you don't have a printer), and you can print it if you'd like. Remember, though, the message will be permanently gone when you press [INPUT]. And that's one of the reasons that there's a message field for appointments: The message remains even after the appointment alarm has gone off.

You can easily get knee-deep in menus while experimenting with the text editor. Feel free to explore these sub-menus; the worst that could happen is that you'll end up with an odd character or two in the display. And again, simply press [←] to erase any extra characters.

The Alpha Menu

We're not limited to numbers and parentheses and such; the text editor gives us access to the full selection of alpha (regular text entry) characters within its menus.

We won't be printing anything in this example, so it doesn't matter if you have a printer. We're showing the message printing feature here because it is so easy use. Press [■][PRINTER] to enter the printer menu. If the calculator beeps and refuses to budge, press [■][MAIN] to exit from any active menu, then press [■][PRINTER].

Keystrokes	Display	Remark
[■] [PRINTER]	0 LIST | STK | REGS | TIME | MSG | TRACE	The printer menu. Now press the fifth [^] key labeled ≣MSG≣.
≣MSG≣	TYPE MESSAGE; [INPUT] ABCDE | FGHI | JKLM | NOPQ | RSTUV | WXYZ	This is the text editor main menu. The menu keys are your typewriter keyboard.
[^]	TYPE MESSAGE; [INPUT] ABCDE | FGHI | JKLM | NOPQ | RSTUV | WXYZ	Press the first [^] key to move to the menu with the first five letters.

TYPE MESSAGE; [INPUT]					
A	B	C	D	E	OTHER

Nothing has been typed yet, so the opening screen is still displayed.

≣A≣

A▒					
ABCDE	FGHI	JKLM	NOPQ	RSTUV	WXYZ

The character is added to the display, and you return to the text editor menu.

Pick a sub–menu, then pick a letter; after the calculator places the letter in the display at the cursor position, the main alpha menu returns. If you enter a sub–menu accidentally, press [EXIT] *once* to return to the previous menu.

Typing Aids and Keys That Beep

When you press number and mathematical operator keys (like [+]), the calculator enters the characters as a typing aids; press [7] and you'll get a 7 character in the display. That is, it doesn't try to make calculations of these keystrokes (remember, we're using the 27 as a text editor now, not a calculator).

For example, [÷] enters the division sign, [■][Y^X] enters the exponentiation symbol ^ (often called a carat), and [■][E] enters the letter E. However, [■][π] (among several others) just beeps. When you are in Solver editing a formula many of these other keys become typing aids, but more on that later when we introduce Solver.

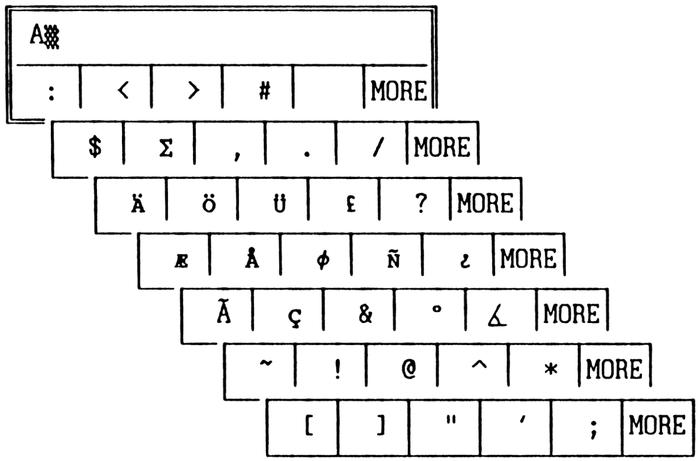
Entering a Space

Sub–menus with only four letters (≣FGHI≣, ≣JKLM≣, ≣NOPQ≣, ≣WXYZ≣) also contain a space character (the fifth [⌃] key). To type a space, simply press ≣FGHI≣ then the ≣ ≣ key.

Other Characters

Once you press one of the alpha menu keys (it doesn't matter which one), the sixth [⌃] key in each sub–menu, marked ≣OTHER≣, is a passage to other special characters. Press ≣OTHER≣ and the calculator shows a sub–menu of these characters.

Cycle through the menu of other characters menus with ≣MORE≣. There is no "LESS" key to back-up; if you miss a character, you'll have to go all the way around. The seventh time you press ≣MORE≣, the display will roll back around to the first other character sub-menu. Return to the alpha menu by pressing [EXIT] *once.*



The most often needed characters are in the first sub-menu. When you find the character you need, simply press the menu key and the calculator adds the character to the line at the position of the cursor. A few characters are missing; the Japanese Yen character (¥) and the PI symbol (π) are gone, but the British pound (£) is there.

Reviewing the Text Editor Keys

Keystrokes	Reaction
[←]	Erases a single character to the left of the cursor and moves the cursor one position to the left.
[CLR]	Erases the entire edit line, leaving the cursor at the beginning of a blank line.

[INPUT]	Ends text editing and enters the text. In [■] [TIME], the information is entered as a message field; in [■] [PRINTER], it is printed.
[EXIT]	Enters the edit (cursor movement) menu. Press the far right [△] key, labeled ≣ALPHA≣, to return to the alpha menu.
[EXIT] [EXIT]	Pressing [EXIT] *twice* cancels text editing without entering the text.
[■] [MAIN]	Exits the operation completely.
[■] [OFF]	You can turn the calculator off in the midst of things and come back later; whatever you were typing will remain in the display.

Edit Menu

Once you've typed a few characters, you may find a typing mistake a few characters back. Instead of clearing the line and starting over or pressing [←] repeatedly to erase characters back to the position of the mistake, you can back the cursor up over what you've already typed and insert or delete characters.

Press [EXIT] *once*; instead of exiting the text editor, the calculator under– stands that you want to move the cursor, so it presents the edit menu. If you haven't typed anything yet, there's nothing to edit, so the 27 exits the text editor.

Keystrokes	Display	Remark
[EXIT] ≣<--≣ ≣<--≣	ABCD▒ DEL | <<-- | <-- | --> | -->> | ALPHA	Enter the edit menu then push the left cursor key twice.
≣DEL≣	AB▒D DEL | <<-- | <-- | --> | -->> | ALPHA	The cursor is now over the "C". Now press ≣DEL≣ to delete that character.

ALPHA

ABD					
ABCDE	FGHI	JKLM	NOPQ	RSTUV	WXYZ

Now press ALPHA (the far right [∧] key) to return to the alpha menu.

Edit Menu Keys

Keystrokes	Reaction
DEL	Deletes the character at the cursor position. All characters from that position shift left to close the gap. Remember, [←] erases the character to the left of the cursor position.
<<--	Moves the cursor left one screen-full (22 characters).
<--	Moves the cursor left by one character.
-->	Moves the cursor right by one character.
-->>	Moves the cursor right by one screen-full (22 characters).
ALPHA	Returns to the alpha menu.

When You're Done Editing

After entering the last character, you have the choice of canceling the operation or completing it. Press [EXIT] to cancel and the previous menu will return. If you press [INPUT] the line will be accepted by the calculator.

Notes about the Text Editor

- The cursor (▒) marks the position on the line where the next character is placed.
- The text editor is always in "insert mode." Whatever you type is inserted in the line to the left of the cursor.
- Maximum of 22 characters per line in [■][PRINTER] and [■][TIME] menus. If you try to enter more characters, the calculator will beep.
- After you type each character, the alpha menu returns.

- If you make a typing mistake, press [EXIT] *once* to enter the edit menu, then press ≡ALPHA≡ to return to the alpha menu. If you accidentally press [EXIT] a second time, you will leave the alpha menu and possibly lose what you've typed.
- The space character is the blank key available on ≡FGHI≡, ≡JKLM≡, ≡NOPQ≡, and ≡WXYZ≡ sub-menus.
- The 27 won't let you leave the text editor until you press [INPUT] to enter the line or [EXIT] to cancel.
- [▲] and [▼] keys do not work while you are editing the line.
- You can turn the calculator off (or it may turn itself off after 10 minutes of waiting) without loosing your place. When you turn the 27 back on, the text editor will be there waiting.
- When you're in Solver or the message field of an appointment, you can enter the [■][PRINTER] menu for other tasks, but you can't type a ≡MSG≡. That's because the text editor is still active in Solver. Exit the current menu and the ≡MSG≡ menu will again work.

Menus and Variables

Back in chapter five we introduced numeric function menus, whose [▲] keys offer functions used in calculations. We've saved two numeric function menus until now to segue into talking about applications, and eventually about Solver.

Keystrokes	Reaction
[■][CONVERT]	Conversions between degrees/radians, decimal hours/hours minutes and seconds, and polar/rectangular coordinates.
[■][%CHG]	Percent change calculations menu.

Both [■][CONVERT] and [■][%CHG] menus are considered to be part of the numeric function library, but there's one distinction between them and the others in that category: menu variables. The Percent Change menu is entirely menu variables, while the Convert menu uses them sparingly.

A cordial relationship with menu variables is a key to understanding the calculator. Every application, including Solver, depends on these menus to communicate with us. The great thing about menu variables is that they're as easy and understandable as numeric function menu keys.

Figuring the Percentages: The [■][%CHG] Menu

The antithesis of the [■][BASE] menu in complexity, Percentage Change calculations are a variation on division. So why such a pedestrian-sounding feature in an advanced calculator? Because, as usual, Hewlett-Packard engineers have added another twist: menu variables.

The Percent Change menu has three menu variables: ≡OLD≡, ≡NEW≡ and ≡%CH≡ (that's a [▲] key, not to be confused with the [■][%CHG] function key). You use menu variables like registers with one major difference: If you store numbers in two variables, when you press the [△] key for the third the calculator will tell you its value!

Keystrokes	Display	Remark
[■] [%CHG]	2,839 OLD NEW %CH	Punch-up the Percent Change menu. As usual, the calculator line moves to the top of the display.
12 ≡OLD≡ 36 ≡NEW≡	NEW=36 OLD NEW %CH	Store 12 in ≡OLD≡ and 36 in ≡NEW≡. While we're in this menu, the menu line shows variable labels.
≡%CH≡	%CHANGE=200 OLD NEW %CH	Calculate the change between 12 and 36.

The number on the calculator line begins with a label: "%CHANGE=200"; press ≡OLD≡ and the calculator line shows "OLD=12". Whenever you press a [△] key to recall or calculate a variable, the result on calculator line is shown with the variable name. If you rotate the history stack down, you'll see that the second number still has the label, though the labels will go away when you exit the menu or use the number in a calculation.

Now, let's add a bit to ≡%CH≡ and see how it affects ≡NEW≡:

Keystrokes	Display	Remark
[+] 50 [STO] ≡%CH≡	%CHANGE=250 OLD NEW %CH	Add 50 to ≡%CH≡. The expression is evaluated when we press [△].
≡NEW≡	NEW=42 OLD NEW %CH	The calculator solves for the unknown and displays the result.

Like the other numeric menus, [EXIT] returns you to the previous menu.

Calculating with Menu Variables

There's some natural confusion about numeric function menus and menu variables; after all, they look exactly the same until you try them.

Here are some more rules that at first seem tricky, though after you've used them a few times, you can again forget them. As usual, the 27 keyboard works just how you'd like it to, though probably not how you expected it to work.

If the last thing you did was enter a number or partially evaluate a formula:

- Press [RCL] [∧] to copy the menu variable to the calculator line.
- Press [STO] [∧] to copy the rightmost number from the calculator line to the named menu variable. If there's only a single number in the calculator line, that number will be used.
- To complete the calculation and enter the result in the menu variable, press the [∧] key for the variable name.

If the last thing you did was anything but enter a formula:

- Press [RCL] [∧] to recall the current value of the menu variable without first re-evaluating the variable.
- Press a [∧] key to evaluate that variable.

Clearing Menu Variables

Old data can be as much of a problem as mathematical errors. Menu variables retain their value between calculations, even if you exit the menu. To be assured that you're working with current data, enter the menu and press [▒][CLEAR DATA] to reset the menu variables. This resets only the variables in the current menu. Be cautious, though, because it also resets the history stack.

If you're doing chain calculations with results floating on the history stack, be sure to store important numbers in registers before you press [▒][CLEAR DATA].

Comparing Variables and Registers

Variables, like registers, are named locations in calculator memory, and they can often be used like registers. Since they're all made of the same stuff, why make this distinction between variables and registers?

The differences are as much philosophical as anything else. Numbered registers, as we've seen earlier, are independent of calculations. They're there as your personal cubbyholes, and are available regardless of what menu is active; like registers in old-fashioned calculators.

A menu variable is a symbol which represents a value, usually the product of a calculation, like ≣NEW≣ above. You gain access to variables through menu keys, and variables are connected directly to formulas and may change as the formulas are re-evaluated.

Since symbolic menu labels represent the variable in a menu, you can only use these variables while the appropriate menu is in the display.

In short: The units for storage, numbered zero through nine, are generally called registers, everything else is a variable.

Applications Menus

Applications differ from numeric function menus in complexity. Their menus have several screens and a number of features and variables.

Keystrokes	Reaction
[■] [STAT]	The statistics (number list) menu.
[■] [TIME]	Time calculations and appointments.
[■] [TVM]	Time Value of Money (financial calculations) menu.

Stat Lists

The [■] [STAT] menu creates order from lists of numbers, finding "facts and figures" from *live* number lists.

Many calculators perform accumulation calculations, maintaining only a running total, sum of the squares, and the total number of items in the list; perhaps they work with two variables, with another sum of squares, the product of the two, and such. The advantage of this scheme is that it requires three, or at most six registers, so it will work in a minimum of memory on modestly powered calculators. The problem is that you cannot review the list or accurately correct a single entry.

The 27, on the other hand, maintains the entire list; you can review, edit, even sort the list, and generally extract more information from *live* data. In fact, you can keep multiple named lists in the calculator.

Besides using the built-in statistical formulas, you can use lists with Solver. Two special functions in Solver return the number of items in the list (SIZES) and the value of a single item (ITEM).

Stat List Keys

Keystrokes	Reaction
[■] [STAT]	Enter the statistics menu.
[INPUT]	Evaluate the expression on the calculator line and enter the results at the current position in the list.
[=]	Evaluate the expression on the calculator line, but do not enter the results in the list.
[▲]	Move up one line in the list.
[▼]	Move down one line in the list
[■] [▲]	Move up to the top of the list.
[■] [▼]	Move down to the bottom of the list.

A Short Demonstration

Instead of the traditional sample list showing sales at Wally's Widget World, we'll use pseudo-random numbers between 0 and 100. Values for this list were calculated in Solver using the RAN# (random number) and RND (round to number of decimal places) functions, with the following formula:

```
A=RND(RAN#x100 : 2)
```

1:	93.65	11:	18.51	21:	8.36
2:	48.64	12:	48.54	22:	81.86
3:	60.14	13:	33.43	23:	10.12
4:	17.4	14:	73.63	24:	82.81
5:	37.23	15:	54.12	25:	59.87
6:	21.29	16:	14.47	26:	47.75
7:	12.73	17:	80.02	27:	78.2
8:	1.22	18:	68.75	28:	59.61
9:	80.65	19:	36.55	29:	97.05
10:	94.13	20:	58.11	30:	17.01

We won't list keystrokes to enter the entire list (we're trying to keep this book down to lug-able proportions). If you make a typing error, press [▲] to move back up to the item in question and change it; no hassles, no starting over, no accumulated errors! In fact, when you're done entering data, press [■] [▲] to move to the top of the list and check your entries.

Keystrokes	Display	Remark
[■] [STAT] ≡GET≡ ≡*NEW≡	ITEM(1)=? CALC | INSR | DELET | NAME | GET | TOTAL	The Stat menu. Create a new list. The calculator is asking you to enter the first item.
≡NAME≡	TYPE A NAME; [INPUT] ABCDE | FGHI | JKLM | NOPQ | RSTUV | WXYZ	So we don't forget, name the list LIST1. Press [INPUT] when you're done typing.
93.65 [INPUT]	ITEM(2)=? TOTAL=93.65	Enter the first item. The calculator displays the running total and asks for the next item.

We'll skip the middle 2 through 29 items. On to item 30...

Keystrokes	Display	Remark
17.01 [INPUT]	ITEM(31)=? TOTAL=1,495.85	The calculator is asking for the next item. There's already some useful data: TOTAL=1495.85.

To enter new data in the list, simply type the number or a calculation then press [INPUT]. Be sure to end any calculation you do not wish to add to the list with [=]. Remember, in Stat [INPUT] always enters the results of the calculation into the list at the current position; if you're not at the end of the list, this replaces the item.

Stat Menu Keys

Keystrokes	Reaction
≡CALC≡	Go to a second screen to calculate
≡INSR≡	Insert a new zero-value item at the current position in the list. All higher-numbered items move down.
≡DELET≡	Delete the single item on the calculator line. Use [■] [CLEAR DATA] to delete the current list.

NAME	Edit the name for the current list. The name can be up to 22 characters, though like variable names, they will be assigned to labels, so make the first 4–5 characters unique.
GET	Change to another list. The menu switches to list available lists. Press *NEW to create a new list.
TOTAL	Show the list total on the bottom line of the display.

Evaluating Stat Lists: The CALC Key

Continuing our example, first we'll press [EXIT] to return to the Stat menu, then we'll move to the Stat calculations menu for some facts and figures.

Keystrokes	Display	Remark
[EXIT]	ITEM(30)=17.01 CALC | INSR | DELET | NAME | GET | TOTAL	We're back at the Stat main menu.
CALC	1,495.85 TOTAL | MEAN | MEDN | STDEV | RANG | MORE	Enter the calculations menu to extract more information from the list.

Stat Calculations Menu Keys: CALC

Keystrokes	Reaction
TOTAL	The sum of all items in the list. This is the same as the running total displayed while editing the list.
MEAN	Arithmetic mean (average). The sum of the items divided by the number of items in the list.
MEDN	Median. The middle of the list. Returns the value of the item midway between the largest and smallest item.
STDEV	Standard deviation. A measure of dispersion around the mean.
RANG	The Range is the difference between the largest and smallest item in the list. That is, MAX minus MIN. Key label aside, this has nothing to do with the beeper.

MORE MIN	The value of the smallest item in the list.
MORE MAX	The value of the largest item in the list.
MORE SORT	Sorts the list; smallest value first, largest last.
MORE FRCST	Calculations using two number lists.

Our random sample returns a MEAN and MEDN which are quite close:

Keystrokes	Display	Remark
TOTAL	TOTAL=1,495.85	The sum of all items in the list.
MEAN	MEAN=49.86	This is TOTAL divided by the number of items.
MEDN	MEDIAN=51.38	RAN# gave us a fairly smooth sample.
STDEV	STDEV=29.38	
RANG	RANGE=95.83	Item 29 minus item 8 is the smallest.

The Difference Between MEAN and MEDN.

The mean is the average of the list (add it all up and divide by the number of items), while the median is a number falling midway between the extremes (sort the list in sequence and find the item in the middle or halfway between the middle two if there are an even number of items).

Using Numbers From Lists in Calculations

Akin to the other menus, you can perform keyboard calculations while you're editing a stat list, though this is one exception to the rule of [INPUT] and [=] working the same. [=] evaluates the formula on the calculator line, while [INPUT] evaluates the formula *and* enters the result in the list.

Item 30 in our list is shown. Let's recall that item to the calculator line, then add item 29 to it. We're in the Stat menu, so the [▲] and [▼] keys move us through the list; pressing [▲] moves up one item in the list.

Keystrokes	Display	Remark
[RCL] [INPUT] [+] [▲] [RCL] [INPUT]	ITEM(29)=97.05 17.01+97.05	The list is shown on the top of the display, the calculation on the bottom. Press [=] at the end of the calculation.

Notes about Stat Lists

- To sort the current list, press CALC MORE SORT, then press [EXIT] *once.*
- [INPUT] enters the number from the calculator line in the list. If you don't want to enter the result of a calculation into the list, use [=].
- To create a new list, press GET *NEW. Remember to NAME the list if you're planning on keeping several in the calculator at once.
- Replace an item in a list by pressing [▲] or [▼] to move the item you want to the display, then type the number and press [INPUT].
- You can add new data to the end of the list by pressing [■][▼] to move to the bottom of the list, then just typing the number.
- Use [RCL] [INPUT] to recall the current Stat item to the calculator line.
- Choosing 360D or 365D toggles between doing calculations in a 360 day banker's year (with evenly divided 30 day months) and the real world.

Financial Calculations: The TVM Menu

Time Value of Money calculations explain why interest rates make your new car end up costing half again what the dealer promised it would. The two options are simple and compound interest. If you're a borrower, you want simple interest; if you're a lender, you want compound interest. If you're a calculator, you also want compound interest because it's more interesting to calculate, so that's all the [■][TVM] menu works with.

Simple interest is, well, simple: the straight interest charged. A $100 loan at 12% per year would end up costing $112. Then divide that total by the number of payments. Hardly worth the time calculating it for an advanced calculator.

Compounding a loan includes interest on interest. That $100 loan at 12% per year (1% per month) is $101 after the first month. The second month interest is based on the total owed at the beginning of the month; you're paying 1% interest on $101, for a total of $102.01. The problem is figuring your monthly payment.

This section should get you started with (▒)(TVM). The HP-27S Owner's Manual has an extensive discussion of compound interest, and is good reading once you're comfortable with the menu.

Keystrokes	Display	Remark
(▒)(TVM)	12 P/R BEGIN MODE N | I%YR | PV | PMT | FV | OTHER	The TVM main menu. Modes are 12 payments per year, due at the beginning of the period.
12 ≡N≡ 12 ≡I%YR≡	I%YR=12 N | I%YR | PV | PMT | FV | OTHER	12 monthly payments and 12 percent per year interest.
100 (+/−) ≡PV≡	PV=-100 N | I%YR | PV | PMT | FV | OTHER	The starting value of the loan is $−100 (negative since you're borrowing it!).
(▒)(MODES) ≡FIX≡ 2 (INPUT)	PV=-100.00 N | I%YR | PV | PMT | FV | OTHER	That's better! Dollar amounts are easier to read in FIX 2.
0 ≡FV≡	FV=-0.00 N | I%YR | PV | PMT | FV | OTHER	When you're done, you want to owe nothing.
≡PMT≡	PMT=8.80 N | I%YR | PV | PMT | FV | OTHER	Each monthly payment is $8.80.

[■] [SHOW]

```
FULL PRECISION IS:
8.7960977013
```

All values are calculated and stored with full calculator precision.

The ≡OTHER≡ menu offers keys to change the number of payments per year and from begin to end mode:

Keystrokes	Reaction
≡N≡	The total number of payments.
≡I%YR≡	Annual interest rate as a percentage.
≡PV≡	Present value. The value at beginning of the loan term.
≡PMT≡	The dollar amount for each payment.
≡FV≡	Future value. Value at the end of the loan term.
≡OTHER≡ ≡P/YR≡	The number of payments per year. Requires an integer in the range of 1 through 999.
≡OTHER≡ ≡BEG≡	Set payments to occur at the beginning of each period. The calculator shows "BEGIN MODE" at the right of the display.
≡OTHER≡ ≡END≡	Set payments to occur at the end of each period. The calculator shows "END MODE" at the right side of the display.

CHAPTER NINE

Introducing Solver

As powerful as the HP engineers have made the 27, there's just so much available in those menus. Instead of leaving it at that, they gave us Solver so that we can adapt the calculator to our needs and get the most from our investment.

What if you could repeat a complex calculation time and again without re-entering the formula? With Solver, you can translate written equations into automatically performed algebraic equations; convert blackboard logic to keyboard logic.

What is Solver?

Solver is a tool to simplify entering and using repetitious or complex (and error prone) mathematical expressions. It's not a just a way to record key-board calculations for playback. The tasks performed by Solver are editing and maintaining a library of equations, and evaluating those equations.

Solver's not just for programmers, it's for anyone who has a problem to solve which can be expressed as a mathematical formula.

Not all functions used in keyboard calculations work in Solver. In turn, Solver has a number of new functions not available for keyboard calculations. You can't, for instance, use [■] [LAST] or base conversions in Solver, nor would it be practical to use many Solver capabilities from the keyboard. While statistical lists are useful by themselves, with the new functions available in Solver you can squeeze new information from them.

Solver vs Programming

If you aren't familiar with conventional calculator programming then you're probably beginning with an advantage over those old-hands at programming languages; you aren't walking into it with a lot of baggage and preconceived ideas about how (and how hard) things work. Programming has procedures and structures and protocols, which turn a simple formula into pages of calculator instructions.

Solver is to programming what shorthand is to, well, to longhand. Entering data, processing, and displaying results are reduced to the single task of getting the formula right; the calculator takes care of the drudgery for you, setting up menus and labeling results.

Solver isn't like programming, and it isn't meant to be.

The Three Sides of Solver

Solver has three duties: editing, viewing the list of equations, and evaluating ("running") those equations. The [■][SOLVE] main menu is the gateway to these features, including a variation of the text editor to handle the task of entering and modifying formulas. ≡CALC≡ checks the formula for errors and enters the variables in the menu, ready to be evaluated.

Keystrokes	Display	Remark
[■] [SOLVE]	<TOP OF LIST> CALC | EDIT | DELET | NEW	The Solver main menu. Use [▲] and [▼] to browse through the list of equations.

Solver Keys

The menu keys, indeed the whole keyboard, takes on new meanings in Solver. At the main menu (when "<TOP OF LIST>" or the current equation is in the calculator line), you can browse the list of equations and see the first 22 characters of each equation. Each time you use Solver, it remembers where you were in the list during your last edit session.

We use these keys to cruise through the list of Solver formulas (usually called equations). These keystrokes are similar to the keys used when you edit statistical lists, though keep in mind that the Solver list and statistical lists are quite different.

Keystrokes	Reaction
[▲]	When you are at the main Solver menu, [▲] moves up to the previous equation in the list.

[▒] [▲]	Moves to the top of the list of equations. The permanent "<TOP OF LIST>" marker is displayed. You cannot enter anything above this marker; when you create a new equation, it will follow the marker.
[▼]	Moves down to the next equation in the list.
[▒] [▼]	Moves to the bottom of the list. The permanent "<BOTTOM OF LIST>" marker is displayed; when you enter an equation, it precedes this marker.
[▒] [CLEAR DATA]	A very perilous key to press while you're in Solver! This deletes *all* equations; use it with caution! Use ≡DELET≡ from the Solver menu to delete a single formula.

Solver Main Menu Keys

Keystrokes	Reaction
≡CALC≡	Evaluates syntax in the equation and prepares it for calculating. When you are in this menu, the history stack is active and you can jump to numeric function menus as needed.
≡DELET≡	When you no longer need a equation, you can reclaim the memory it consumed by pressing ≡DELET≡; the calculator displays: "DELETE THE VARIABLES?" and the menu keys ask: ≡YES≡ ≡NO≡. If you select ≡YES≡, the calculator erases the variables used by the formula then displays "DELETE THE EQUATION?" and the menu keys: ≡YES≡ ≡NO≡. The problem is, if you say no to deleting the variables, it won't let you delete the equation. If you have another equation using those same variables, the variables will be cleared when you delete a possibly unrelated equation.
≡NEW≡	In Solver, ≡NEW≡ enters the new item in the list following the current object (except if you are at the bottom of the list, labeled "<BOTTOM OF LIST>").
≡EDIT≡	Edits an *existing* Solver formula.

The Solver Editor

Instead of being simply keystroke programmable (meaning that you enter formulas just how you'd use them in keyboard calculations), Solver uses the text editor to simplify editing long formulas and entering functions not on the keyboard. We'll discuss some of these functions; the rest are listed in the back of this book.

If you haven't experimented with the text editor yet, turn back to the chapter called (conveniently enough) "The Text Editor."

Using the Solver Editor

To enter the editor, use the (▼) and (▲) keys to move the equation you want to edit to the calculator line, then press ≡EDIT≡ to call-up the text editor. When you're done editing the formula, (INPUT) replaces that item in the list with the new version of the formula that you just created.

When you press ≡NEW≡ from the Solver main screen, the calculator jumps immediately to the text editor alpha menu instead of the cursor menu. The reason for this is obvious when you stop to think about it: Usually the first thing you'll enter in a equation is a variable name; there's nothing in the line yet to edit, so the 27 is saving you a keystroke by moving directly to the alpha menu.

Keystrokes	Display	Remark
≡NEW≡	TYPE EQUATION; [INPUT] ABCDE | FGHI | JKLM | NOPQ | RSTUV | WXYZ	Enter a formula using text editor keys. Press (INPUT) when you are finished editing the formula.

Solver Editor Typing Aids

Solver's version of the text editor has more typing aids than used in the (■)(TIME) and (■)(PRINTER) menus. These new keystrokes enter function names from the numeric function menus without leaving the editor.

You have two choices when entering a function: Either type in the name, or jump to a function menu from the editor and press the (△) key for the function; the function won't be evaluated, but the function name is entered in the formula. For example, press (■)(PROB) to enter the probability menu (a wonderfully ethereal name for a menu) then ≡N!≡ for factorial; the cal–

culator types "FACT(" at the cursor position, then *immediately* returns to the editor.

Keystrokes	Display	Remark
[shift] [PROB] ≡N!≡	FACT(▒ ABCDE | FGHI | JKLM | NOPQ | RSTUV | WXYZ	The typing aid is entered at the cursor position then the alpha menu returns.

The left parenthesis is included in the typing aid because functions in Solver always need them. As you can see, the 27 isn't exactly "keystroke programmable;" in fact, "FACT(" is just a series of characters on the line that you can edit any way you'd like. And if you press [shift] [E], you get an everyday E, not just the exponentiation symbol.

Notes About the Solver Editor

- You can edit and play with existing equations in any way, equations in the Solver list aren't changed until you specifically tell the calculator to replace them.
- After you edit an equation, all of the variables used by that equation are reset to zero, even those shared with other equations.
- Function keys are typing aids. For instance, [shift] [1/x] enters "INV(".
- There's no 22–character limit for equations. If an equation exceeds the display width, the calculator shows a set of ellipses (...) on either the left or right end (or both ends) of the displayed line to indicate the additional data.
- If you press [EXIT] to leave Solver after modifying an equation, the 27S asks "SAVE THIS EQUATION?" and the menu shows ≡YES≡ ≡NO≡ ≡EDIT≡. While the 27's full–line prompts are a far cry from earlier calculators, this one's still a bit cryptic. What it means is "do you want to save new the edits you've just made, or leave things as they were before you started?" Obviously, that's a bit long for the dis–play. You can press ≡NO≡ or [EXIT] to lose any changes, ≡YES≡ to replace the old formula with the new one, or ≡EDIT≡ to continue editing the equation.

Solver as a Data List

We haven't talked about beeps and error messages because Solver lets you enter whatever you want to as a equation. As long as there's enough memory to fit the "equation" in the calculator, you can type anything! As long as you don't try to use CALC on it, you have a phone number list. This illustration shows the phone number for HP Calculator support in Corvallis, Oregon, entered without extra spaces to save memory.

Keystrokes	Display	Remark
	HP(503)7572004 CALC | EDIT | DELET | NEW	HP Calculator Support phone number listed in Solver.
CALC	INVALID EQUATION CALC | EDIT | DELET | NEW	Beep! The calculator calls up the editor and places the cursor at where it thinks there's an error.

You can enter appointment memos, printer messages, and Stat lists names as long as 22 characters. In Solver, as with keyboard calculations, there is no restriction on line length, and formulas may be quite long or complex. The calculator shows ellipses (...) when the line grows past 22 characters.

As you enter formulas or notes, keep track of available memory every now and again by pressing (shift)(MEM). Each character you type costs one byte. Memory for Solver, variables, statistical lists, and appointment memos comes from the same bucket; while it'll never overflow, checking it often offers some assurance that it'll never run dry.

How you order your list is a personal decision, though usually it's best to locate it at the bottom of the Solver list, following all *real* equations. If you have a great number of items in the list, it's easier to find the first or last item (by pressing (shift)(▲) or (shift)(▼)) than one toward the middle. Deciding on alphabetical order, order of frequency, related groups, or just entering them as you go is also a consideration when entering real equations.

Remember, when you press NEW to enter a new formula, it is entered in the list following the equation on the calculator line. To insert a line before the item in the example above, place the line preceding "HP(503)7572004" which may be "TOP OF LIST" or another formula, on the calculator line and then press ≣NEW≣.

If you decide to use Solver for phone lists and want some semblance of security, add an entry or two labeled "<BOTTOM OF LIST>" and enter your important list following it. This should convince the casual browser that there's nothing else there; that is, if they've found their way this far. Just as useful for security might be to move to a dull menu when you leave your calculator unattended. There's not much fun to be found in, for instance, the [▒][BASE] menu.

Solver Equations

With function names spelled–out, Solver equations are easy to read, though they look nothing like keyboard calculations. First, the simplest possible equation:

```
A
```

That's it. This formula doesn't look like much, but it does do something: When you press ≣CALC≣, the calculator evaluates this formula; recognizing that A refers to a Solver variable name, it creates the variable (if it doesn't already exist), then displays a menu containing the variable name.

Keystrokes	Display	Remark
≣CALC≣	0 A	The 27 automatically cre–ates menu labels for each variable in our formula.

We can use [RCL]≣A≣, [STO]≣A≣, or just plain ≣A≣ as we do with, for instance, the [▒][%CHG] menu. The difference is that this is a formula we created and can change at will.

Like numeric function menus, the calculator line is active for regular key–board calculations; you can press a menu key to enter a function menu, then press [EXIT] to return to this equation. Back to the [▒][%CHG] menu analogy, let's write a Solver version of that menu:

```
((NEW-OLD)÷OLD)x100=%CH
```

As you can see, closing parentheses for all expressions are required in Solver and, as usual, we'll get back to that subject later. Now evaluate the formula to see a reasonable facsimile of the [■][%CHG] menu:

Keystrokes	Display	Remark
[CALC]	`0` `NEW │ OLD │ %CH`	A variation on the [■][%CHG] menu, though we created it in Solver.

Solver Variables

Variable names in each menu are listed as they appear in the formula, from left to right. Though if a variable appears several times in the formula, it'll still only appear once in the menu. If there are too many variable names to fit on one screen, the calculator adds a [MORE] menu key and lists the remaining variables on the second screen.

The final variable assignment (assigning to [%CH] in this case) can be on the left or right, and occasionally doesn't happen at all. In the example above we placed the variable assignment on the right. Next comes the same formula, but the variable assignment is on the left, which may be more comfortable for some people, but the resulting menu is less pleasing:

```
%CH=((NEW-OLD)÷OLD)x100
```

Keystrokes	Display	Remark
[CALC]	`0` `%CH │ NEW │ OLD`	Variable labels in the menu changed as we changed their position in the formula.

Creating and Clearing Variables

The variable [NEW] was created automatically when we pressed [CALC] to evaluate the equation. When an equation contains a new variable name, the calculator automatically creates the variable when you evaluate the equation. There are three ways to clear variables and recover the memory they consume:

- When you edit a formula which uses the variables.
- When you press [■][CLEAR DATA] with the formula on the calculator line and press ≡YES≡ to confirm deletion.
- When you press ≡DELET≡ and answer ≡YES≡.

Sharing Variables between Equations

The name ≡NEW≡ could be used in any number of Solver equations, and in each it would retain the same value until you or the next equation changed it. In this way, you can evaluate an equation, then pass results to the next equation just by using some of the same variable names in each formula.

Besides sharing information between equations, a second (and often just as important) reason to use the same variable names in several equations is to save memory. Since there's only one variable, there's only one variable–worth of memory needed for storing it.

Solver variables are separate entities from numeric function and other application menu variables. ≡NEW≡, ≡OLD≡, and ≡%CH≡ are different variables from those in [■][%CHG], even if they have the same name. Depending on your purposes, this is either an advantage or a problem. Since you can't read or change, for instance, [■][TVM] variables in Solver, there's no possibility of them being accidentally changed. Alas, you can't change these variables on purpose either.

Naming Variables

There are two considerations when naming variables: Making them readable, and making them fit on the menu labels. Foremost, the name should be *mnemonic*, expressing the purpose of the variable.

> **mnemonic** (ni–mon'ik) *adj.* Aiding or designed to aid the memory. –n. A mnemonic device such as a rhyme or formula.

≡%CH≡ is certainly a mnemonic label, though it can be confusing to read in the formula since "%" is traditionally an algebraic operator.

While a name may be up to ten characters, in general, only the first four or five characters will fit on a label, including the blank dot between each character. Each menu label is 21 display dots wide; the narrowest characters are three dots, while the remainder are either four or five dots:

WIDTH OF CHARACTERS IN MENU LABELS	
4 Dots:	G K N Q Σ ? Ä ö £ Æ Å ϕ Ñ ¿ Ç @
5 Dots:	M W # $ & ~

From this table, you can see that a label "MONEY" wouldn't fit because the width exceeds 21 dots. More of a problem is that while it's mnemonic, "MONEY" is vague; "COST" is both more descriptive and fits on a label.

Notes about Variable Names

These restrictions are both for our benefit and the calculator's benefit when interpreting expressions. If variable names aren't clear, the calculator could misinterpret a variable name and either fail to understand the formula at all, or evaluate it incorrectly.

- Variable names may contain a maximum of 10 characters.
- If several names are longer than 5 characters, be sure the first 5 characters in each are different to assure that menu labels are unique.
- Cannot begin with a number, though they may contain numbers.
- Names cannot contain a space.
- Names cannot contain mathematical operators (+, ÷ ...).
- The period is allowed if it is not the first character.
- Names should not be the same as function names.

Using Calculator Variables With Solver

First the good news: While we're in Solver evaluating an equation, we can [STO] and [RCL] the results of formulas in the ten numbered calculator registers. The bad news is that there's no way for Solver formulas to directly read or write calculator variables. If you need to read a register into Solver, recall the value to the calculator line, then press the menu variable name. Remember that the history stack is active when you're evaluating a formula in Solver.

[RCL][0] [STO] NEW

Notes about Solver Variables

- Variables retain their value, even when you use other equations or exit Solver.

- If two or more equations use the same variable name, the same variable is used in each equation.
- A variable doesn't exist until you have evaluated (pressed ≣CALC≣) an equation referencing the variable name at least once.
- Whenever you edit an equation, the variables are cleared. Variables which are shared with other equations are also cleared.
- Use descriptive names for variables.
- Keep variable names short enough to be readable in the menu.
- Variables used in other menus are not shared by Solver. The [■][%CHG] variable ≣OLD≣ and the [■][TVM] variable ≣N≣ are not available in Solver.
- Be careful about naming variables. To avoid syntax errors and confusion when editing equations, don't use function names (such as MAX or Σ) for variable names.

Naming Equations

Another twist before we get too deep into Solver: Equation labels. These are descriptive names at the beginning of formulas, separated from the formula by a colon. You can give an equation any label at all, and when the 27S evaluates the formula, it ignores it entirely. The calculator has no interest whatsoever in these labels; they are just for our reference, so you can use them or not at your discretion.

```
LABEL:«variable» = «argument»
```

Like menu variables, equation names cannot contain mathematical operators and symbols (like ÷ or +) or spaces. If you forget to add a label (or you change your mind), you can edit the equation and enter or change the label whenever you want to.

Remarks

Instead of (or in addition to) an abbreviated cryptic label at the beginning of the formula, consider entering a descriptive remark on the line preceding equations. The line might contain comments or instructions about the equation which follows.

A standard symbol at the beginning of the line preceding the formula designates it as a comment. HP programming languages often use an exclamation mark (!) to designate a comment.

```
! NOTE ABOUT EQUATION
LABEL:«variable» = «argument»
```

There is a catch to this system for remarks: don't try to evaluate the remark as an equation. Nothing awful will happen, it just won't work.

Solver Syntax

Imagine translating a page written in a foreign language that you don't know how to read, say French, into English. With a French / English dictionary at your side, word-by-word the text finds its way into English. Part way through the page some French expressions will be repeated and you'll be able to write the English equivalents without turning to the dictionary. When you've finished the job, you'll go back to the top of the page and start smoothing out the rough parts because French doesn't translate directly to English. A final check eliminates any remaining faux pas.

Press CALC and Solver goes through the translation of an English-like algebraic expression into 27ese. Which is a preferable to making us write the formula in 27ese to begin with. As the calculator looks through the formula, it's more disciplined than we are, because it always uses the dictionary and never makes any assumptions about what we mean, avoiding the faux pas from the outset.

One joy of learning Solver is coming up with a long convoluted formula, replete with functions and parentheses, and discovering that the HP-27S can understand it.

Solver Functions

Many Solver functions aren't assigned to keys; you'll have to type them in. In fact, that's the main reason we spent a whole chapter talking about text editing.

With functions, there's now another source for parentheses in the formula. In addition to parentheses to control order of operator priority, function arguments (also called parameters) are enclosed within a pair of parenthesis. The argument itself may be the result of another function; this is called nesting functions.

Each function either requires a matching set or no parentheses at all. How do you know what arguments the function requires? We could cite rules and historical precedence, but there are only four functions requiring no arguments: CDATE, CTIME, PI, RAN#. The number of arguments required by other functions varies, but when you understand the purpose of the function, the number of arguments is often self-evident.

Most trigonometric functions work with a single value. MIN and MAX return the smaller or larger of two numbers, so obviously they require two arguments each. For the more specialized functions (read: least intuitive) you'll probably frequently refer to the back of this book.

Notes about Solver Syntax

- The x operator is required for multiplication instead of implied multiplication using parentheses. Use Ax(B÷C) instead of A(B÷C).
- Parentheses are used both to alter the order of operator execution and to enclose arguments for functions.
- When Solver doesn't understand the formula, it will quickly show the error "INVALID EQUATION" then call the editor with the cursor pointing at the first character in the formula which the 27 didn't recognize as part of a valid formula.
- Don't enter commas in numbers.
- Evaluation of an equation is not halted by the "OVERFLOW" error, though obviously the results will be suspect.

Operator Priority in Solver

This table is expanded a bit from the mathematical precedence table the 27S follows when doing business as a calculator. It's larger because the calculator has more capabilities in Solver than keyboard calculations.

SOLVER OPERATOR PRIORITY	
( )	Expressions in parentheses.
ABS RADIUS SIN . . .	Functions.
y^x	Exponentiation (^).
x ÷	Multiplication and division.
+ −	Addition and subtraction.
NOT	
AND	
OR XOR	

Pairs of Parentheses

The most common reason the 27 will refuse to run a Solver equation is missing or misplaced parentheses. When you press CALC to evaluate an equation, the 27 doesn't add missing closing parentheses as it does in calculator mode; you have to insure that the expression is complete.

The 27 doesn't require the minimum number of possible pairs of parentheses. If you enter extra sets (and providing that they match), extras are ignored in the calculation. Since Solver doesn't alter the written formula, it only interprets it, any extra parentheses will remain in the formula. As evidenced by this little formula, there are often a number of closing parentheses at the right of the formula, and that's usually where the mismatch errors show up.

```
A=((B + ( C )))
```

With our little 22-character window on the world, it's occasionally tough to make parentheses match. Inspecting a print-out of the formula will help if you just can't seem to get beyond the "INVALID EQUATION" message.

An easy way to insure that you have the correct number of parentheses is to count them; working from left to right, add one for every left parenthesis, subtract one for every right parenthesis. If you end up with zero, it's likely that the parentheses match and the error is elsewhere. A positive number means that it's likely that closing parentheses are missing; a negative value generally means that you've added too many closing parentheses. This doesn't insure that parentheses are placed correctly, just paired accurately.

CHAPTER TEN

Putting Solver to Work

Think of ≡CALC≡ as a key to enter a custom menu; as [shift] [%CHG] enters the Percent Change menu, pressing ≡CALC≡ from Solver enters our menu. There's a bit of a lag first while the calculator verifies the equation, looking for errors and allocating variables if necessary.

When Solver Calculates an Equation

Press [▲] for an unknown and Solver reads your equation, organizing it following algebraic rules (isolating the unknown), and tries to calculate an immediate solution to the unknown. What happens when the unknown can't be directly found? Solver doesn't give up, it quickly calculates two guesses at the solution, then narrows the search until it finds a solution.

If it can't find an answer using those two guesses, Solver estimates two new guesses and has another go at it. The process is to evaluate the left and right sides of the equation, then calculate left minus right (looking for zero). It repeats the process for the other estimate, as the two close–in on each other.

Guesses

You can help Solver and possibly speed execution by entering a guess for the unknown, or better yet two guesses, simply by typing the guess then pressing [STO] [▲] for each estimate. If you enter only a single estimate, Solver will supply the second; enter no estimate at all and Solver will supply both, and depending on how well it does with those guesses, execution may continue for some time. A bad guess is better than none at all. During the search, Solver displays the two guesses it's trying out at the moment:

```
VARIABLE:70.6199398344    +
VARIABLE:70.6199398343    -
```

The "+" and "–" at the right of the display are the sign of left minus right for these guesses. Solver may take a few minutes to calculate an unknown (park the calculator on the corner of your desk and take a break). You can press any key (except [■]) to suspend the search for the solution and see how it's progressing. Press [CLR] (or [←]) and the calculator will show the closest solution found so far. To continue the search, press the [^] key for the unknown, and Solver will get back to work.

It's possible that Solver won't come up with an exact result (left minus right isn't exactly zero) using the estimates, though through several iterations, it'll come up with close estimates for both sides. The calculator will beep and display the estimates for left and right:

```
LEFT :6.99999999998
RIGHT:7.00000000000
```

The limit to the accuracy of Solver's guesses is the limit of the accuracy of the calculator; often the best answer will lie between the two estimates (which are generally very close). If the estimates aren't close, or if you're sure that a solution can be found (and that there really is only a single solution), enter new guesses and try again. If things don't go well, Solver shows "BAD GUESS. PRESS [CLR] TO VIEW". Be sure to press [CLR], because it doesn't work if you press [←] in this case.

Conditional Expressions: The Big IF

At one time computers were machines that could make decisions, while calculators could not. That is, react differently, doing one thing under one set of circumstances, and another under different circumstances. The IF function gives Solver that capability. Drawing the line between calculator and computer is getting harder all the time.

Solver is quite English-like, making for readable formulas. This is especially true with the syntax and usage of IF. If the test expression evaluates true, then the expression following the colon is performed, otherwise it's ignored and the expression after the *second* colon is evaluated. That's the same syntax as the previous sentence.

Equation	Reaction
`X = IF(`	The IF function.
`«Test» :`	If this test expression evaluates true,
`«Expression» :`	then evaluate this expression,
`«Expression»`	otherwise evaluate this expression.
`)`	

Logical Comparisons

Also called ***Boolean logic***, after George Boole, an English mathematician of the 19th century, logical comparisons return either a true (if the test evaluates true) or false value (if the test does not evaluate true). 7>1 is a logical comparison which evaluates true because 7, indeed, is larger than 1. This illustration will *always* yield 3; X will never return 9E37.

```
X = IF( 7>1 : 3 : 9E37 )
```

Boolean comparisons only work within the structure of the IF function. The following is a logical truth table, listing the result returned by each comparison.

BOOLEAN TRUTH TABLE				
A evaluates to B evaluates to	1 1	0 0	0 1	1 0
A > B	False	False	False	True
A < B	False	False	True	False
A = B	True	True	False	False
A >= B	True	True	False	True
A <= B	True	True	True	False
A <> B	False	False	True	True
A AND B	True	False	False	False
A OR B	True	True	True	True
A XOR B	False	False	True	True
NOT A	False	True	True	False

Comparison	Remarks
>	Test if argument A is larger than argument B. IF(A>B : C : D)
<	Test if argument A is smaller than argument B. IF(A<B : C : D)
=	Test if two arguments are equal. IF(A=B : C : D)
<>	Argument A does not equal argument B. IF(A<>B : C : D)
AND	If both test A and test B are true. IF(A=T AND B=T1 : C : D)
OR	If either test A or test B or both are true. IF(A=T OR B=T1 : C : D)
XOR	Exclusive OR. If either test A or test B is true, but not both. IF(A=T XOR B=T1 : C : D)
NOT	If test A is not true. The opposite of the argument. IF(NOT A=T : C : D)

Examples of IF

Expression	Remarks

`A=IF(`
`MOD(4 : 4 )`
`> 5 :`
`C :`
`D )`
	If the expression
	modula(4:4) (the remainder of 4÷4)
	is greater than 5
	then return the value of variable C
	otherwise return the value of variable D.

`A=IF( NOT`
`MOD(4 : 4 )`
`> 5 :`
`C :`
`D )`
	If the expression does NOT evaluate true...
	This is the opposite of the test above.

`A=IF(`
`A>B OR`
`B<C :`
`D :`
`E )`
	If either (or even both) of the expressions evaluates true...

One of IF's greatest strengths can sometimes be a problem: you can make incredibly long, complex, convoluted, confusing formulas. In fact, it's often preferable to use other functions to simplify formulas and speed execution. For example, these formulas are equivalent:

```
A=IF( C>D : C : D )
A=MAX( C : D )
```

Both formulas serve the same purpose (to return the largest of the two variables C and D), though the second version takes fewer keystrokes, less memory, and a bit less time for the calculator to figure out. When possible, look through the library of functions for MIN, MAX, or MOD, and leave IF for the hard jobs.

Notes about IF

- If the expression evaluates true then the formula following the colon is evaluated, otherwise evaluate the formula following the second colon.
- The conditional expression may be any logical comparison which evaluates to a true or false result.
- Any expressions and formulas, including additional IF functions, may be conditionally executed in an IF expression.
- If the calculator shows "INVALID EQUATION" when you use an algebraic equation as a conditional, and you are sure your equation is correct, try inserting a plus before the left parentheses before the left parenthesis. Change `IF((A+B)x6 < C : ...` to `IF(+(A+B)x6 < C : ...`
- The conditional expression must be complete. You cannot use IF(A:C:D), meaning "IF A is not zero then...", you must use IF(A<>0:C:D).
- Boolean comparisons only work within the IF structure. You cannot use "A=A+(A>0)".
- IF has the same usage as the IF... THEN... ELSE structure in most programming languages.

A Menu of Formulas: The Big S

S, which stands for "Solving for," is a variation of IF which lets us create one menu with two equations; an obscure talent it would seem at first, but handy when its use becomes clear.

The menu of variables is presented, then when the user (programmers like to call us "users") enters data in the menu variables, the calculator chooses the first formula if the specified variable (or variables) key was pressed, or the second formula if it was not.

Expression	Remarks
`IF(`	Use one of two formulas depending on which menu
`S(«Variable»):`	key is pressed. If we are solving for «Variable»
`«Equation» :`	then use this equation.
`«Equation» )`	Otherwise use this equation.

The equations are arranged to evaluate to zero, though the "=0" isn't required. If M&Mx3.8=CAL then the expression is organized as M&Mx3.8−CAL=0, and it's entered in the formula as M&Mx3.8−CAL. This formula tests for either M&M or CAL.

```
IF( S(M&M) OR S(CAL) : M&Mx3.8-CAL : CHIPx1.4-CARB )
```

Notes about S

- The variable must appear in the equation, not only the S() statement, for it to appear in the menu.
- When checking for several possible variables, syntax is critical. IF(S(A) OR NOT S(B)... is correct, while IF(S(A) AND NOT S(B)... does not work.

Iteration: The Big Σ

Σ is a fascinating variation on a theme found in most languages: looping (or iteration). Basically, it means that you may write a formula and tell the calculator to evaluate it a number of times. If you are working with a number of items from a Statistical list, Σ may be the only way, short of evaluating an equation a number of times or creating a phenomenally long formula, to work with several elements from the list.

Equation	Reaction
`X = Σ(`	The Σ function. Executes the expression a number of
`«Counter» :`	times, storing the counter in this variable starting
`«StartValue» :`	at this value and incrementing the counter variable
`«EndValue» :`	until the counter variable exceeds this value.
`«StepSize» :`	Incrementing the counter by this amount each time.
`«Expression»`	Evaluate this expression each time and sum the result.
`)`	

Σ is not the conventional looping structure, which evaluates an expression a given number of times, and that's all. True, the formula on the right is evaluated, however the function returns sum of *all* iterations. And since we've named the loop counter, we can use that variable name in the equation.

```
A=Σ( COUNT : 1 : 10 : 1 : COUNT )
```

The variable COUNT is the name of the loop counter; COUNT begins at 1, and is incremented until it passes 10, the step value (the amount to add to COUNT each time) is also 1. The formula (in this case, just the value of the counter variable itself) is evaluated ten times. After the final loop the function returns 55, or the sum of 1+2+3+4+5+6+7+8+9+10.

Change the step value and COUNT will increase by 2 in each loop. In this case, the Σ function returns 25, the sum of 1+3+5+7+9.

```
A=Σ( COUNT : 1 : 10 : 2 : COUNT )
```

We'll use Σ again in the next section to read items from a Stat list.

Notes about Σ

- The last (mathematical expression) argument is calculated once each iteration, and the function returns the sum of all calculations.
- The loop counter variable is not entered in the user menu because it will be immediately changed by the starting value.
- Any or all of the arguments required by Σ may the result of mathematical expressions. Only the final argument is evaluated for each iteration; the other arguments are evaluated only once when the function begins.
- Specify a start value larger than the end value and a negative step size and the counter will decrement until it reaches the lower value: X=Σ(COUNT:10:1:−1:COUNT).
- If the start value is greater then the end value, the mathematical expression won't be evaluated at all.

Using Lists with Solver

Two Solver functions are designed just for working with Stat lists. SIZES returns the quantity of items in a named list, and ITEM returns the value of a single item in the list. SIZES requires just the list name in parentheses, ITEM requires both the list name and the item number. This illustration reads the second to the last item in LIST1.

Equation	Reaction
`A=ITEM(`	Look–up the second to last item in LIST1.
`LIST1 :`	In case the list only has one item, use
`  MAX( 1 :`	MAX to insure that subtracting 1 doesn't end up 0.
`    SIZES(LIST1)`	Find the size of the list.
`    -1 )`	Subtract 1 from the list size.
`)`	The end of the structure.

This next illustration reads an item from a Stat list based on the value of another item in the list. Say the first item in the list points to another item (or even an item in another list), we would read the first item, then use that value as an index into the list.

Equation	Reaction
`A=ITEM(`	Look–up the item in
`LIST1 :`	LIST1, pointed to by the first
`  ITEM( LIST1 :`	item in LIST1.
`  1 )`	
`)`	

This third illustration is a basic structure to read a number of items from a Stat list. Like pressing ≡TOTAL≡ while in the ▦ STAT menu, this formula returns the total of items in a list, though this version is much slower, tally–ing only about 16 items from the list per second. It uses Σ to loop through the list.

Equation	Reaction
`A=Σ(`	Find the total for items in LIST1.
`COUNT :`	COUNT is the counter variable.
`1 :`	Start at the first item in the list.
`SIZES(LIST1) :`	End at the last item in the list.
`1 :`	Increment the counter by one each iteration.
`ITEM( LIST1 :`	Find an item in LIST1
`      COUNT )`	COUNT points to the current item in the list.
`)`	The end of the loop.

If the list doesn't exist, SIZES returns zero and ITEM returns an error. The formula will halt with "SOLUTION NOT FOUND". Unless you're absolutely sure of the number of items in a list, formulas should always use

SIZES to verify list size. Trying to read an item beyond the end of the list is a sure way to cause Solver to halt evaluating the formula with an error that's hard to track down (after all, the formula *looks* correct).

Several Stat lists may be referenced in a single expression; this formula uses SIZES on both lists and executes for only as many items as appear in the shorter of the two lists.

```
A = Σ( COUNT : 1 : MIN( SIZES( LIST1 ) : SIZES( LIST2 ) ) : 1 : SQ(
ITEM( LIST1 : COUNT )) x SQ(ITEM( LIST2 : COUNT )) )
```

Because of a design bug in many 27S', be sure to specify only integers for ITEM. For more information and some cautions about using ITEM, see the reference section in the back of this book.

Avoiding Errors in Solver

What we have here is a failure to communicate... It pays to remain aware of possible data errors when you write Solver equations. Frequent oversights when we write Solver equations are missing parameters for a function, or not separating parameters with colons.

Often we use ABS, MIN, MAX, RND, TRN, and occasionally FP or IP to insure that data is within a specified range.

```
ALOG( ABS(«argument1») ) + ALOG( ABS(«argument2») )
```

This returns the logarithm of the unsigned product of the two arguments.

Notes about Preventing and Detecting Solver Errors

- Review the syntax requirements for functions in the back of this book.
- Many functions require numbers to be within a specific range. If a number is outside of the range, Solver will halt with "SOLUTION NOT FOUND" displayed. When you execute these functions directly with bad data, the error is "INVALID INPUT." For example, ACOS requires the argument to be in the range of minus one through one, and FACT can only deal with integers in the range of zero through 253. Any value outside of those ranges, and Solver will halt, regardless of how well you've constructed everything else.
- Try as much of the expression as possible in immediate mode to test for accuracy and operator precedence.

- Have you insured that parentheses match? Count opening and closing parenthesis to make sure they come out even.
- Compare the printed listings to the formula.
- Are arguments (parameters) separated by colons (:). It's easy to accidentally enter commas, especially when translating formulas from other sources.
- Have you entered a comma in a large number (10,000)?
- A popular (and entertaining) pastime during computer programming sessions is called "beta testing," a process of trying to make a program fail. Try your equations in as many ways possible to assure that it will work in every situation.

Notes about Organization

- Define the goal. What is the exact purpose of the equation? Try to state the purpose of the equation in a single sentence; if you can't, it's probably too complex.
- Divide the job into modules. Remember, equations can share variables, so use separate equations and move from one to the next as you approach the final solution for your data. A big equation is a bunch of small formulas.
- Plan on paper before entering Solver. The more time you spend planning, the less time you'll have to spend re-writing formulas.
- Write the formula completely first, without concern for speed or memory conservation. Then when it's running, make it more aesthetically pleasing.
- Don't re-invent the wheel. If a similar equation has already been done, use it as a starting point.
- Print work in progress often. Be sure to include the time and date. You'll get a better feel for what the equation does from the listing. Keeping the old listings will monitor the growth of the project, and allow you to backtrack if things go awry with later changes.
- Name variables so that someone else will be able to understand the purpose of the equation and how to use it from the menu labels. Perhaps you are the only one who will ever use the equation, but several months later, will you remember what they mean?
- Have we planned for maintenance? The equation will go through revisions and may be the basis for another solution.

- Keep it readable. When possible, add spaces for clarity.
- Conserve memory when you are sure that the formula won't be compromised. For instance, 1E3 takes less memory than 1000.
- Find the simplest way to write an expression. Both LOTO=MAX(1:INT(RAN#x39)) and LOTO=RND(RAN#x39:0) perform the same task.
- If you can afford the memory, add a remark describing the equation on the immediately preceding line.
- Expect the worst and plan for it. Things happen. People press the wrong keys, and the use of your equation may change.
- Make sure you have a permanent written copy of the equation.

Finding Solutions for Solver

You probably already have equations in mind, but it's often practical to adapt Solver equations from other machines. Programs originally written for mainframe computers have been modified for desktop machines, and others written on small handhelds often find their way to the computer at the office.

It's a practical way to do things, and not always as difficult as it sounds. Since digital machines are based on similar technology, there are bound to be similarities. Some of the design features of the 27, in fact, is based on earlier portable computers, which in turn were based on desktop machines.

Keep an eye out for technical journals and magazine articles which include examples or theory, even if you've never heard of the programming language and the program listings look like prehistoric cave paintings. File them away, and one day when you find a need to solve a problem that's new to you, the work of those author's will provide a starting point. Even if the examples were written in Cobol in 1960 or on an HP–21 in 1975 there's always something to be learned. Beware, though, that you may be learning from the mistakes (represented in the articles as truths) of those authors.

Programming Languages

Most calculator and computer programming languages are algebraic. Surprisingly, you can often extract the logic of a formula (or the complete algorythm) without being familiar with the programming language. The first task is to find the difference between the program structure and the logic used in formulas. Often you can throw away the program structure and look for the underlying algorithm. Languages share similar structures:

- *Initialization.* This code is usually located physically at the beginning of the program. Initialization allocates memory for variables and changes certain machine settings. Since variables are allocated automatically by the 27 and the calculator is ready to use without changing things around, you can usually ignore this section, though it may provide a listing of variable names used in critical formulas.
- *Data input.* Program languages usually stop and ask the user for data, then assign the data to variables. Again, the 27 does away with this.
- *Data processing.* This is the meat of the program. Locate these sections (there may be several in programs which do a number of things) and try to extract the logic.
- *Reporting.* This is the end of the program run. Using keywords like DISP or PRINT or printf, they read the results of data processing and display a formatted report of the results. Since HP-27 variables are always available in the menu, this part of the program is unnecessary. Be careful that processing isn't combined with reporting; for example, instead of "X=2+Y, DISP X", the program may use "DISP 2+Y".

While ostensibly similar, algebraic languages often differ when interpreting operator priority (mathematical precedence). Exponentiation is generally evaluated before functions.

Since the 27 is algebraic and it was designed by Hewlett-Packard, there are a number of similarities between Solver and HP BASIC. In BASIC, formulas usually refer to a number of variables, and the variable assignment is on the left.

Most languages use a comma (,) to separate arguments in functions, not the colon (:). Other common symbols used by many languages are:

<table>
<tr><th colspan="2">SYMBOLS</th></tr>
<tr><th>Symbol</th><th>Meaning</th></tr>
<tr><td>/</td><td>÷ Division.</td></tr>
<tr><td>*</td><td>x Multiplication.</td></tr>
<tr><td>\</td><td>IDIV integer division.</td></tr>
<tr><td>#</td><td><> Does not equal; especially HP BASIC.</td></tr>
<tr><td>"</td><td>Beginning or end of a string.</td></tr>
<tr><td>!</td><td>NOT (in the C language). Also != (not equal).</td></tr>
<tr><td>|</td><td>OR (in the C language).</td></tr>
</table>

A single equals sign may designate a variable assignment; two equals signs (==) may refer to a logical comparison in some languages.

Languages generally include string functions. A string is any series of characters, such as a line of text written by the 27's text editor, enclosed within quotation marks. Therefore, a string function does something with a string; UPRC$, for instance, turns characters into upper case. In BASIC, strings and string functions end with a dollar sign. Since the 27S is devoid of string functions (and it doesn't miss them at all), these programs will be exceedingly difficult to translate. It is a number machine, after all.

Spreadsheets

An electronic spreadsheet is a program which represents data and formulas as a table with columns and rows. The intersection of each column and row is called a cell. Each cell can contain data or a formula, and the formulas can reference other cells in the spreadsheet.

So why are we talking about spreadsheets? The simplest Solver formulas are quite similar to formulas in electronic spreadsheet programs running on personal computers. If you decide to adapt spreadsheet formulas, it will probably be necessary to have the owner's manual for the program.

With Solver, any number of variables may be used in the formula, and the calculator will solve for any unknown. The spreadsheet is similar in that other cells (items in a list) may be referenced in the formula. However the variable assignment is on the left, and the values of other cells referenced by the formula are never changed by the formula. In the spreadsheet, the cell names themselves are referenced instead of variable names, so the value returned by the formula is the value of the cell.

Formulas will either reference a single cell or a list. Using lists with the 27 is similar to referencing a single column in a spreadsheet. Since the spreadsheet is most often used for tabular data, you'll usually find functions which sum a list. A list in a spreadsheet could be arranged across a row in-stead of a column, though in practice it's rarely done.

This table lists how a formula would reference the fourth item in the third column in a spreadsheet file. In the case of the 27, it's a list called C.

DATA REFERENCES	
Command	Environment
ITEM(C:4)	HP-27S.
C4	Lotus 1-2-3 and VisiCalc.
R3C4	Microsoft Multiplan.
[3,4]	WorkBook71.

Most spreadsheet formulas refer to the column letter then the row number; Microsoft Multiplan is the exception; cell references list row number then column number. A formula may also reference a range of values by name; the name is a label given to that series of columns and rows.

The sigma function (Σ) is rarely found in spreadsheets, relying instead on recalculation. Recalculation is the process of reading and re-evaluating each formula in the file. Formulas in spreadsheets are evaluated a single time for each recalculation, though some programs allow you to set the program to recalculate the file for a specific number of iterations or until some condition is met.

There's often a function called S which has usually has *nothing at all in common* with the 27's S function. While the syntax may be similar, the spreadsheet version is usually a string function.

When converting formulas from Lotus 1-2-3 (or one of the many compatible programs), the most obvious difference is that 1-2-3 requires a commercial at sign (@) at the beginning of each function name.

Equation Solvers

An equation solver is a program, not unlike the 27 Solver, running on a desktop computer. They share many of the capabilities of the 27 and similar syntax. Many of these programs plot functions on a printer or pen plotter. Equation solvers are useful when working with large data samples.

Although these programs run on large powerful desk-bound machines, you'll not find them to be several-thousand-dollars-worth faster than the 27S, because these are general purpose machines running an application program, while the 27 was designed from birth to solve equations. Among currently available equation solvers are Eureka by Borland International, MathCAD by Mathsoft, Inc., and TK!Solver by Universal Technical Systems Inc. Each of these programs costs more than the HP-27S!

HP Algebraic Calculators

The closest calculators to the 27S are the HP-18C Business Consultant (now discontinued), HP-19B Business Consultant II, the HP-17B, and HP-22S Scientific. The 18C is an older design whose version of Solver has fewer features, but formulas should transfer to the 27S without change.

HP has a series of "Consultant" solutions books for use with these business machines; the examples and solutions should readily transfer to the 27. The HP-22S Scientific also has solutions books, but the abbreviated version of Solver on that machine limits the sophistication of the solutions.

HP RPN Calculators

The HP-41 (and it's relatives) speaks a variation of Reverse Polish Notation, or postfix syntax. Converting from these machines again requires an understanding of the language and the purpose of the program.

Converting from RPN is a major undertaking (as it probably was in the first place when the program was translated to RPN). When it's available, you'll probably find more insight in the documentation than the program itself. Unless you are thoroughly familiar with RPN calculators, converting from them may prove to be more difficult than starting from scratch.

The Printer

By this point we hardly need to stress the value of making permanent copies of important formulas and data. The HP82240A thermal printer is a simple and reliable device, relying on low-intensity infrared signals instead of a wire. In fact, HP is selling the printer to third parties for use with their equipment.

Sharing the Printer

If you'd like a printer, but can't justify the expense, consider sharing the printer with fellow HP calculator user. This printer works with other HP calculators, including the HP-18C, 19B, 28C, and 28S. And with an extra adapter (HP82242), the HP-41 series of calculators can also use the printer. Beware that while they're similar in appearance to the 27S, two HP calculators do not work with the printer: the HP-22S and HP-32S.

If you are within "shooting distance," several calculators can use the printer at the same time. Occasionally you'll be playing dueling calculators; when there is a dispute over which calculator sent the signal, it's the printer that loses--data that is. At times calculators seem to get along like crabs and lobsters; after printing each line, the 27 returns the printhead to the left of the line, while some other HP calculators send it to the right.

Printer Menus

Two menus and one special key control the printer:

Keystrokes	Reaction
[■] [PRINTER]	The Printer menu selects groups of items to print.
[■] [PRT]	Send the calculator line to the printer.
[■] [MODES]	The AC adapter toggle is in the MODES menu.

Printing

[■][PRT] is a powerful key; it prints whatever is on the calculator line. In Solver it prints the current equation. In Stat, it prints the current item in the list. In Time, it prints the current time and date. At most other times, it prints the number on the calculator line.

From the [■][PRINTER] menu, you can print lists and notes and archival copies of anything you've entered into the calculator.

Keystrokes	Display	Remark
[■] [PRINTER]	`0` `LIST STK REGS TIME MSG TRACE`	You can enter the printer menu from anywhere. Press [EXIT] *once* to leave this menu.

Printer Menu Keys

Keystrokes	Reaction
LIST	Sends the entire list to the printer. In Solver, all formulas are printed. In Stat, the current Statistical list is printed. In TVM and other applications, a list of menu variables is printed.
STK	Prints the contents of the history stack.
REGS	Prints the contents of registers 0–9.
MSG	Prints a message of up to 22 characters.
TRACE	Traces keyboard calculations. Each key you press and the result of every operation will be sent to the printer. This slows down most calculator operations.

Printer AC Adapter

Hewlett–Packard is a company famous for high resolution pen plotters, ink jet printers, and laser printers. Lineage aside, without fresh batteries or an AC adapter, printouts from the HP82240A, or any thermal printer, can look quite blotchy. The [■][MODES] menu sets-up the calculator to work with a printer connected to an AC adapter:

Keystrokes	Display	Remark
(■)(MODES)	SELECT DISPLAY FORMAT FIX | SCI | ENG | ALL | ./, | MORE	The MODES menu first screen.
≡MORE≡	SELECT MODE D/R | BEEP | PRNT | | | MORE	The MODES second screen.
≡PRNT≡	PRINTER: AC ADAPTER D/R | BEEP | PRNT | | | MORE	This is a toggle; each time you press ≡PRNT≡, the display changes between "AC ADAPTER" and "NO AC ADAPTER".

With AC adapter means that the calculator sends data to the printer faster; the calculator doesn't know or care if the printer actually has an AC adapter connected. If you've set it to the faster setting and you are running the printer on batteries, be sure to use fresh batteries to make sure data isn't lost when the calculator gets ahead of the printer.

The HP82240A isn't picky about what AC adapter you use. Polarity isn't an issue, and it really prefers 9–12 volts. After trying it with a half dozen AC adapters, the only adapter it balked at was one putting out about 3.5 volts; it worked, but the pale-gray image crawled slowly from the printer.

Notes about Printing

- The AC adapter does not recharge batteries. If you use rechargeable batteries, be sure to use a separate charger.
- The printer owner's manual lists special characters and printer control codes which are ignored by the HP-27S, but supported by some calculators.
- Setting the printer to trace slows most operations down to a crawl whether you are using a printer or not. Only use trace mode when the printer is on.

CHAPTER THIRTEEN

More Than You Probably Want To Know

The first generation personal electronic calculator was a gangling mechanical beast, as likely to walk off the desk as to finish chugging through a multiplication problem (they were the mechanized counterpart to counting with your fingers). Since that time the personal calculator has gone thorough a number of incarnations.

This new generation calculator is actually a computer disguised as a calculator. If you will, it's a computer running a calculator program, and that's part of the magic of it all.

Lineage of the 27S can be traced to the HP–35 of 1972, the world's first handheld scientific calculator. A higher model number doesn't designate a later or more sophisticated machine; after all, the first handheld HP business calculator was the HP–80, with a single–line LED display (Light Emitting Diode, the little red lights which slurp battery power at a prodigious pace). In the beginning, the model number designated a series based on a common calculator shell; the original 10–series, for example are all physically similar. In recent years, model numbers have been less consistent.

Interim machines included the 67 and 97, powerful scientific machines with built–in card readers. These were followed by the 34C (where Solver originated) and 38C handhelds, which were quite popular, especially in the real estate trade, until September of 1981 when the tiny 12C and 15C calculators, with a single line LCD (Liquid Crystal Display) and horizontal keyboard redefined the small handheld calculator.

In 1979 HP introduced the HP–41C, probably the most successful calculator ever created, supported by an enormous variety of software and accessories. The HP–41 is still available, but it has a single–line display and is slower and has less memory than the 27S. Between the introduction of the 15C and the 27S, Hewlett-Packard offered the BASIC language HP-71B handheld computer. Surprisingly (at first speculation), the closest kin is the HP Business Consultant, with folding keyboard and a 4-line display.

Inside the HP-27S

The unifying element of this family tree is the microprocessor, the brain (or, more philosophically, the soul) of the machine. Being innovative and technically oriented people, Hewlett-Packard engineers developed their own microprocessor for the 27S, with greater mathematical precision and versatility than the processor found in, for example, the average desk-bound personal computer. Based on the Saturn processor found in the HP-71B portable computer, the 27 is optimized for high accuracy math and low power consumption.

For those with a technical bent, the Saturn processor has eight byte registers and four bit (half byte) address lines. So, depending on how you look at it (or who you ask), it is either a 4-bit or 64-bit computer; either way it's a unique and powerful machine. Add to this the 64K Operating System and CMOS RAM (Complimentary Metal Oxide Semi-conductor), a low power consumption type of chip RAM Memory.

There's a keyboard buffer which lets you type ahead, faster than the calculator can handle the commands, without losing keystrokes. Move to a menu with an undefined key (such as the rightmost [^] key in the [■][TIME] menu) and quickly press and release that menu key 15 times; the calculator will continue to beep up to 15 times after you're done.

Each element in the low-power LCD display is updated at 1/32nd second intervals. With the right lighting, you can see this rippling effect as the calculator writes each pixel (a computer acronym for picture element) from top to bottom.

Calculator User's Interface Evolution

The evolution toward the menu driven calculator hasn't always been smooth. HP has taken a number of side roads; The HP-35 had a single shift-key, with operations printed on the face of the key and keyboard bezel. As the number of functions grew, both the number of keys, and the number of shift-keys in particular, increased until the HP-34C, which had three multi-colored shift-keys to match the four functions per key! It was easy to spend afternoons looking for functions that you knew were there somewhere. The other extreme was the HP-75, a powerful handheld computer with no typing aids at all, just a straight typewriter keyboard; you typed the commands (providing you could remember them all), then pressed a key labeled "RTN."

Total keystrokes is secondary to the number of brain clock–cycles it takes to translate the task at hand to arcane mechanical monkeying. While our heads can handle concurrent processes, pressing keys should probably remain a background process. To that end, came the menus.

Mathematical Precision

Since the calculator has 8–byte registers, calculations are performed with at least 12–digit mantissa and 3–digit exponents, regardless of how the numbers are displayed. Several functions are calculated internally with up to 15–digit mantissa, by splitting numbers between two CPU registers.

Bytes

In plain English, a byte is the smallest *usable* piece of information a calculator or computer can represent; often a single character like "A" or "%". Numbers aren't stored as characters but in machine readable form called Binary Coded Decimal, they always require eight bytes. For this reason the value zero takes the same amount of memory as 12.6E237. When it comes time to show us the number, the calculator translates it into characters we can read.

About this Book

This book was edited and formatted with Microsoft Word 4.0, and printed using Softcraft Fancy Word version 3.02. Headline fonts are Bitstream Dutch; fonts for body text (including this text) and examples (keystroke and display) were designed by the author.

HP-27S Command Index

How to use this List

Function and command names are printed in large type followed by a short description. If a function can be used in keyboard calculations, the appropriate keystrokes are printed to the right of the description. When no shortcut keystrokes are listed, the function can only be used in Solver.

Solver syntax for functions is demonstrated in a box on the line following the command name. When an operation is not usable in Solver or it is a command key (such as [INPUT]), the example box is empty.

Arguments for functions can themselves contain references to other functions. For simplicity, we'll use «argument» to represent any argument which evaluates to a number.

Examples assume ALL (floating point) display mode and degrees angular mode. Enter ALL mode by pressing [■][MODES] then press the ALL menu key. Switching between radians and degrees is in the same menu; press MORE for the second menu, then D/R to toggle either radians or degrees mode. The "RAD" display annunciator lights when the calculator is in radians mode; there's no annunciator for degrees mode.

ABS Absolute Value. [■][PARTS] ABS

```
«variable» = ABS( «argument» )
```

Returns the absolute value (magnitude). That is, if the argument is negative, ABS makes it positive; positive values are unchanged.

Argument Evaluates To:	ABS Returns:
ABS(1)	1
ABS(−4)	4

ACOS Arc cosine. [ACOS]

```
«variable» = ACOS( «argument» )
```

Arc cosine. Returns the principal of an angle having a cosine equal to its argument. The inverse of COS.

Argument Evaluates To:	ACOS Returns:
ACOS(.75)	41.4096221093
ACOS(−.5)	120

The argument must be in the range of −1 through 1. ACOS will cause a "SOLUTION NOT FOUND" error if the argument is out of range in Solver. "INVALID INPUT" is returned for a keyboard calculations.

ACOSH Hyperbolic arc cosine. [HYP] ACOSH

```
«variable» = ACOSH( «argument» )
```

Hyperbolic arc cosine. The inverse hyperbolic cosine of the argument.

Argument Evaluates To:	ACOSH Returns:
ACOSH(1)	0
ACOSH(2.5)	1.56679923697

The value of the argument must be equal to or greater than one.

ALOG Common (base 10) antilogarithm. [10^x]

```
«variable» = ALOG( «argument» )
```

Common (base 10) antilogarithm.

Argument Evaluates To:	ALOG Returns:
ALOG(1)	0
ALOG(10.1)	1.00432137378

The value of the argument must be greater than zero.

AND — Logical And.

```
«variable» = IF(«conditional_argument» AND «conditional_argument» :
  «expression» : «expression» )
```

Logical (Boolean) AND is used with IF. See IF and page 81.

ANGLE — Polar angle.

[shift] [CONVERT] MORE ∠/

```
«variable» = ANGLE( «argumentX» : «argumentY» )
```

∠ Polar coordinate for (x,y) rectangular coordinates.

Argument Evaluates To:	ANGLE Returns:
ANGLE(1 : 1)	45
ANGLE(2 : 1)	26.5650511771

ASIN — Arc sine.

[shift] [ASIN]

```
«variable» = ASIN( «argument» )
```

Arc sine. Returns the principal of an angle having sine equal to its argument. Uses the current angle mode setting.

Argument Evaluates To:	ASIN Returns:
ASIN(1)	90
ASIN(.25)	14.4775121859
ASIN(−.25)	−14.4775121859

The argument must be in the range of −1 through 1.

ASINH — Arc hyperbolic sine.

[shift] [HYP] ASNH

```
«variable» = ASINH( «argument» )
```

Returns the inverse hyperbolic sine (arc hyperbolic sine) for the argument.

Argument Evaluates To:	ASINH Returns:
ASINH(1)	1.44363547518
ASINH(−15)	−3.40230664548

ATAN Arctangent. [shift][ATAN]

```
«variable» = ATAN( «argument» )
```

Returns the arctangent of x in the range of −90 to +90. Uses the current angle mode setting. The inverse of TAN.

Argument Evaluates To:	ATAN Returns:
ATAN(15)	86.1859251657
ATAN(1)	45

ATANH Inverse Hyperbolic Tangent. [shift][HYP] ≡ATNH≡

```
variable» = ATANH( «argument» )
```

Returns the inverse hyperbolic tangent of the argument.

Argument Evaluates To:	ATANH Returns:
ATANH(0.75)	9.72955074528E−1
ATANH(−.5)	−5.49306144334E−1

ATANH arguments must be greater than −1 and smaller than 1.

BASE Enter base conversions menu. [shift][BASE]

The base menu performs conversions between Decimal/ Hexadecimal, Octal/ Binary. Base conversions are not available in Solver. See page 43.

CDATE Current date.

```
«variable» = CDATE
```

Current date. Returns the current date in the format MM.DDYYYY (month day year) or DD.MMYYYY depending on the current date format setting. Remember, the value will differ depending on the date format. Set the date format in the TIME menu by pressing [shift][TIME] then pressing the ≡M/D≡ menu key. CDATE requires no argument.

CDATE is the equivalent of pressing TODAY in the [■][TIME] CALC sub-menu. Since you can't enter the [■][TIME] menu from Solver while you're editing a formula, there is no typing aid equivalent to CDATE.

Today is Sept 24, 1988:	CDATE Returns:
in MM.DDYYYY	9.241988
in DD.MMYYYY	24.091988

CDATE is useful in formulas using the DATE function to calculate number of days relative to today.

CLR Clear the calculator line. [CLR]

Reset the calculator line to zero or erase text line. See page 25.

COMB Combinations. [■][PROB] C X,Y

```
«variable» = COMB( «argumentX» : «argumentY» )
```

Finds the number of possible combinations of x items taken y at a time. See also PERM.

Argument Evaluates To:	COMB Returns:
COMB(12:3)	220
COMB(5:2)	10

CONVERT Enter Convert menu. [■][CONVERT]

The convert menu includes conversions between degrees/radians, decimal hours/hours minutes and seconds, and polar/rectangular coordinates. Also changes between degrees and radians with [■][CONVERT] MORE D/R. See DEG, RAD, HRS HMS, XCOORD, YCOORD, RADIUS and ANGLE.

COS Cosine. [COS]

```
«variable» = COS( «argument» )
```

The cosine of the argument. Uses the current angle mode setting. The inverse of ACOS.

Argument Evaluates To:	COS Returns:
COS(60)	.5
COS(75)	2.58819045103E-1

COSH Hyperbolic cosine. [■] [HYP] [COSH]

```
«variable» = COSH( «argument» )
```

COSH returns the hyperbolic cosine of the argument. Uses the current angle mode setting.

Argument Evaluates To:	COSH Returns:
COSH(1)	1.54308063482
COSH(10)	11,013.2329201

Furnish a value in the range of negative through positive 1151.98569368.

CTIME Current time.

```
«variable» = CTIME
```

Returns the current time in H.MMSS (hours minutes and seconds) format. Returns the number in 24–hour format regardless of the current time display format. CTIME requires no argument.

Argument Evaluates To:	CTIME Returns:
9:31 A.M.	9.3100
4:15 P.M.	16.1500

DATE(d1:d2:cal) Days between dates.

```
«variable» = DATE( «date1» : «date2» : «calendar_year_type» )
```

Calculates the number of days between the dates date1 and date2. Uses the current date format (set in [■][TIME] ≡SET≡ menu). Set the date format in the Time menu by pressing [■][TIME] then pressing the ≡M/D≡ menu key. DATE is the Solver equivalent of the [■][TIME] ≡CALC≡ menu. The third argument (calendar_year_type) is a number representing the type of calendar year as follows:

1 Calendar year of 365 days.
2 Calendar year of 365 days, but ignores leap years.
3 Use a calendar year of 360 days made up of 12, 30-day months.

Argument Evaluates To:	DATE Returns:
DATE(10.191987: 10.191988: 1)	366
DATE(10.191987: 10.191988: 2)	365
DATE(1.011989: 09.041989: 1)	246

Use CDATE to recall the current date for use in formulas with the DATE function. DATE understands leap-years (when you use 1 for calendar year type), and eliminates the rounding problems normally associated with calculating with 365.25 day years.

DEG Radians to degrees. [■][CONVERT] ≡>DEG≡

```
«variable» = DEG( «argument_in_radians» )
```

Converts the argument in radians to degrees.

Argument Evaluates To:	DEG Returns:
DEG(.5)	28.6478897565
DEG(.02)	1.14591559026

E — Enter exponent. [■][E]

```
«variable» = «mantissa»E«exponent»
```

Regardless of the number display format, you can enter very large or small numbers by entering the mantissa part, pressing [■][E] to enter the exponent symbol, then the exponent of the number.

To enter a negative exponent (for very small numbers), enter the mantissa part then press [■][E] to enter the E. Then press [−] to enter a minus sign. Finally, enter the exponent.

Argument Evaluates To:	Exponent Returns:
9E4	90000
1E22	1.E22

You can also use E as a variable name in Solver, but it's best to avoid it because of the likelihood of confusion.

EXIT — Exit from a menu. [EXIT]

See pages 17 and 42.

EXP — Exponential. [■][eˣ]

```
«variable» = EXP( «argument» )
```

Exponential. Natural antilogarithm *e* to the power of the argument. "*e*" is 2.71828182846. Use exponentiation (^) to calculate using other bases. EXP is the inverse of LN.

Argument Evaluates To:	EXP Returns:
EXP(1)	2.71828182846
EXP(2.5)	12.1824939607

EXPM1 — Natural exponential minus 1.

```
«variable» = EXPM1( «argument» )
```

Natural exponential minus 1. $e^x - 1$.

Argument Evaluates To:	EXPM1 Returns:
EXPM1(1)	1.71828182846
EXPM1(2.5)	11.1824939607

FACT Factorial (Gamma). [■][PROB] N!

```
«variable» = FACT( «argument» )
```

Factorial (Gamma) function; usually displayed as "N!" though Solver uses FACT. Returns the factorial from the integer argument in the range 0–253.

Argument Evaluates To:	FACT Returns:
FACT(4)	24
FACT(36)	3.7199332679E41

FACT will cause a "SOLUTION NOT FOUND" error if the argument is not an integer or is out of the range 0–253. From the keyboard, the error is "N! N<0 OR N NONINTEGER". If the number is too large (larger than 253), the error is "OVERFLOW" for keyboard or Solver, and the value 9.99999999999E499 is returned.

FP Fractional part. [■][PARTS] FP

```
«variable» = FP( «argument» )
```

Returns the fractional part of the argument. See also IP and RND.

Argument Evaluates To:	FP Returns:
FP(1.95)	.95
FP(−4.75)	−.75

HMS Degrees to decimal hours [■][CONVERT] >HMS

```
«variable» = HMS( «argument» )
```

Hours Minutes Seconds. Converts decimal hours (degrees) to H.MMSS (D.MMSS) format. HMS is the opposite of HRS.

Argument Evaluates To:	HMS Returns:
HMS(13.55)	13.33
HMS(22.45054)	22.7515

HRS Decimal hours to degrees. [■][CONVERT] ≡>HR≡

```
«variable» = HRS( «argument» )
```

Converts the argument in H.MMSS (D.MMSS) format to decimal hours. HRS is the opposite of HMS.

Argument Evaluates To:	HRS Returns:
HRS(13.33)	13.55
HRS(22.7515)	22.45054

HYP Enter Hyperbolic menu. [■][HYP]

This menu includes hyperbolic and inverse hyperbolic operations. See SINH, COSH, TANH, ASNH, ACOSH, and ATANH.

IDIV Integer division.

```
«variable» = IDIV( «argumentX» : «argumentY» )
```

Integer division. Returns the *integer part* of the quotient of argumentX ÷ argumentY.

Argument Evaluates To:	IDIV Returns:
IDIV(5:2)	2
IDIV(9.9:4)	2

IF

```
«variable» = IF( «conditional_argument_evaluates_true» : «do_this» :
  «else_do_this» )
```

A logical (boolean) expression. IF evaluates the conditional argument then, if the argument evaluates true (non-zero), the expression following the colon is evaluated. If the argument is false (returns zero), the expression following the second colon is evaluated. See page 80.

INPUT Enter data. [INPUT]

```
```

Terminates any operation. In keyboard calculations, can be used interchangeably with [=]. See page 14.

INT(x) Integer part.

```
«variable» = INT( «argument» )
```

Returns the integer portion of the argument. See also IP and RND.

Argument Evaluates To:	INT Returns:
INT(1.05)	1
INT(3.99)	3

INV Reciprocal; 1/x. [1/x]

```
«variable» = INV( «argument» )
```

INV is the Solver equivalent to the [1/x] key in keyboard calculations.

Argument Evaluates To:	INV Returns:
INV(.5)	2
INV(8)	.125

IP Integer part.

[▓] [PARTS] ≡IP≡

```
«variable» = IP( «argument» )
```

Returns the integer part of the argument. See also FP and RND.

Argument Evaluates To:	IP Returns:
IP(1.95)	1
IP(−4.75)	−4

ITEM Value of an item in a list.

```
«variable» = ITEM( «list_name : «item» )
```

Returns the value of an item in a statistics list. Specify a stat list name and the item number. Use the SIZES function to determine the number of items in the list.

Argument Evaluates To:	ITEM Returns:
ITEM(GROUP1 : 11)	Eleventh item in the GROUP1 list.
ITEM(SAMPL : 4)	Fourth item in the SAMPL list.
ITEM(L3 : SIZES(L3))	Last item in list L3.

Comments about ITEM

ITEM will cause a "SOLUTION NOT FOUND" error if the list does not contain the specified item number. If the item number is not an integer, it will be rounded up to the next whole number.

The first release of the HP−27S (at least through May, 1988) has an error in the ITEM function. The function checks that the integer part of the argument is in the proper range *before* it rounds the number to an integer; more accepted practice would round the argument first. The problem surfaces at both extremes of possible parameters. If the argument is under one but greater than zero, for example 0.9, it is considered invalid because it appears to be zero.

At the other extreme, if the integer part of the number is equal to the number of items in the list, but rounds up to one greater than the number of items, the calculator does not return an error. In fact, what happens is ITEM reads the next eight bytes *beyond the last item in the list*; those eight bytes will likely not be a valid number so the answer will not be predictable. For example, if you specify item 28.7 in a list containing 28 items, there's no telling what ITEM may return.

In this second case, ITEM will return a value which may not be a real number, and may cause a problem if you in turn pass that value to another function. Since functions expect valid numbers, passing that value could cause further problems.

Argument Evaluates To:	ITEM Returns:
ITEM(T1 : 0.9)	An error message.
ITEM(T1 : SIZES(T1)+.9)	Unpredictable value

The easiest solution is to only use integers, though a more practical answer is to use MIN, MAX or IP to qualify data before ITEM sees it.

LAST Recall last result. [■][LAST]

See page 28.

LN Natural (base e) log. [■][LN]

```
«variable» = LN( «argument» )
```

Returns the natural (base *e*) log of the argument. LN is the inverse of EXP.

Argument Evaluates To:	LN Returns:
LN(.5)	−0.69314718056
LN(200)	5.29831736655

The argument must be greater than zero or an error is returned.

LNP1 Natural log plus 1.

```
«variable» = LNP1( «argument» )
```

Returns the natural log of the argument plus one.

Argument Evaluates To:	LNP1 Returns:
LNP1(.5)	4.05465108108E-1
LNP1(199)	5.29831736655
LNP1(0)	0

LOG Common (base 10) logarithm. [■][LOG]

```
«variable» = LOG( «argument» )
```

Returns the common (base 10) logarithm of the argument. The argument must be greater than zero.

Argument Evaluates To:	LOG Returns:
LOG(.5)	-3.01029995664E-1
LOG(200)	2.30102999566

The argument must be greater than zero or an error is returned.

MAX Larger of two arguments.

```
«variable» = MAX( «argument» : «argument» )
```

Returns the larger of the two arguments. See also MIN.

Argument Evaluates To:	MAX Returns:
MAX(2.2 : 2.3)	2.3
MAX(-1 : 0)	0

MEM Show available memory. [■][MEM]

Displays amount of currently unused memory. Also displays available memory as a percentage of total memory. See page 13.

MIN Smaller of two arguments.

```
«variable» = MIN( «argument» : «argument» )
```

Returns the smaller of the two arguments. See also MAX.

Argument Evaluates To:	MIN Returns:
MIN(2.2 : 2.3)	2.2
MIN(−1 : 0)	−1

MOD(x:y) Modulus.

```
«variable» = MOD( «argumentX» : «argumentY» )
```

Modulus. Returns the remainder of x÷y.

Argument Evaluates To:	MOD Returns:
MOD(5 : 2)	1
MOD(2 : 5)	2
MOD(2 : 0)	2
MOD(0 : 2)	0

While division is used, MOD does *not* cause a division by zero error if the second argument evaluates to zero; it instead returns zero. Some HP computer languages cause an error if the second argument is zero.

MODES Enter modes menu. [■][MODES]

The MODES menu controls calculator settings like number format and Degrees/Radians mode. See page 18.

NOT Logical NOT.

```
«variable»= IF(NOT «conditional_argument» : «expression» : «expression»)
```

Logical NOT is used with IF. See page 81.

OFF Turn off calculator. [■][OFF]

Pressing [■][OFF] will turn off the calculator at any time. Press [ON] to later continue from *exactly* where you left off.

OR Logical OR.

```
«variable» = IF(«conditional_argument» OR «conditional_argument» :
  «expression» : «expression» )
```

Logical OR is used with IF. If either of two arguments evaluates true then expression is true. See page 81.

PARTS Enter parts menu. [■][PARTS]

```
```

The parts menu has functions to extract parts of a number. See ABS, FP, IP, RND, and page 42.

PERM Permutations. [■][PROB] P=X,Y

```
«variable» = PERM( «argumentX» : «argumentY» )
```

Finds the number of possible permutations of x items taken y at a time. See also COMB.

Argument Evaluates To:	PERM Returns:
PERM(12:3)	1,320
PERM(5 : 2)	20

PI Pi. [■][π]

```
«variable» = «argument» x PI
```

Returns the constant PI (3.14159265359), the ratio of a circle's circumference to its diameter. PI requires no argument.

Argument:	PI Returns:
PI	3.14159265359

PRINTER — Enter printer menu. [shift key] [PRINTER]

The PRINTER menu sends information to the printer. Use [shift key] [PRT] to send a single item to the printer. See page 95.

PROB — Enter probability menu. [shift key] [PROB]

The probability menu performs combinations, permutations, factorials, and random numbers. See COMB, PERM, FACT, and RAN#.

PRT — Print calculator line. [shift key] [PRT]

See page 95.

RAD(x) — Degrees to radians. [shift key] [CONVERT] >RAD

```
«variable» = RAD( «argument» )
```

Converts the argument (in decimal degrees) to radians. See also DEG.

RADIUS — Radius. [shift key] [CONVERT] MORE R

```
«variable» = RADIUS( «argumentX» : «argumentY» )
```

Returns R polar coordinate for rectangular coordinates. Uses the current mode (degrees or radians).

RAN# Random number. [■][PROB] ≡RAN#≡

```
«variable» = RAN#
```

Returns a fractional random number in the range of 0<x<1. Since the random number is calculated from a seed you can specify, you could use the RAN# in, for example, an encription formula, then later replay the sequence by again using that seed before decoding.

The pseudo-random number sequence is calculated using a seed from the current time of the calculator's clock. You can reset the seed by storing a new value in the RAN# variable. Enter zero to use a new clock-derived value, or a non-zero value to use your own seed. Press [■][PROB] to enter the probability menu, then [0] [STO] ≡RAN#≡ to enter the seed. See pages 58 and 89.

RCL Recall from a register. [RCL]

Recall the contents of a register or menu variable. See page 34.

RND Round. [■][PARTS] ≡RND≡

```
«variable» = RND( «argument» : «number_of_decimal_places» )
```

Round the argument to the specified number decimal places. When the number of decimal places is in the range 0–11 it rounds up. If the number of decimal places is in the range of –1 through –12, it rounds significant digits. When you execute RND in keyboard calculations, the number is rounded to the current number display format. If you're using RND from the keyboard, set the display format in the [■][MODES] menu. Don't confuse this function with RAN#. See also IP and FP and page 58.

Argument Evaluates To:	RND Returns:
RND(7.123456 : 4)	7.1235
RND(7.123456 :–4)	7.1230

RND will cause a "SOLUTION NOT FOUND" error if the number of decimal places is not an integer or outside of the range of –12 through 11.

S   Solve For.

```
IF( S(«variable») : «expression» : «expression» ) = 0
```

Used with IF to select one of two arguments to evaluate. See page 83.

SGN   Sign of the argument.

```
«variable» = SGN( «argument» )
```

Returns the sign of the argument. If the argument is positive, SGN returns the value one (1), negative values return minus one (–1), and if the argument is zero, SGN returns zero.

Argument Evaluates To:	SGN Returns:
Any positive number	1
Zero	0
Argument is less than zero	–1

SHOW   Show full numeric precision.   [■] [SHOW]

```
```

Display the full numeric precision of a number. See page 15.

Σ   Sigma, The Sum Function.

```
«variable» = Σ( «counter» : «start_value» : «end_value» : «step_value» :
  «expression» )
```

Designed for use with lists, but useful for any mathematical expression requiring multiple iterations. Σ repeats any mathematical expression for a predetermined number of times. Specify a counter variable name, starting value, end value, size of the step, and any expression. The value in the counter begins at the starting value (start_value), and increments by the value of the step (step_value) until it exceeds the end value (end_value). The function returns the sum of all iterations of the expression.

The loop counter variable is not assigned to a menu key, though you can (and usually do) use it in the expression. See page 84.

SIN Sine. [SIN]

```
«variable» = SIN( «argument» )
```

The sine of an angle. See also ASIN.

Argument Evaluates To:	SIN Returns:
SIN(30)	.5
SIN(75)	9.65925826289E-1

SINH Hyperbolic sine. [■] [HYP] SINH

```
«variable» = SINH( «argument» )
```

Hyperbolic sine. Uses the current angle mode setting.

Argument Evaluates To:	SINH Returns:
SINH(.5)	5.21095305494E-1
SINH(5)	74.2032105778

SIZES Number of items in a list.

```
«variable» = SIZES( «list_name» )
```

Returns the number of items in the specified list. SIZES is often used before ITEM and the summing function (Σ) to determine the list size to be sure that those functions aren't halted by an error.

Argument Evaluates To:	SIZES Returns:
ITEM(L3 : SIZES(L3))	Last item in list L3.
IF(SIZES(L3)=0:. . .	Returns zero for an empty list.

SIZES will cause a "SOLUTION NOT FOUND" error if the specified list does not exist. Returns zero if the list is empty.

SPFV Future Value.

```
«variable» = SPFV( «interest%_per_period» :
  «number_of_compound_periods»)
```

For a $1 payment, at interest%_per_period and number_of_compound_periods, returns the future value.

SPPV Present Value.

```
«variable» = SPPV( «interest%_per_period» :
  «number_of_compound_periods»)
```

For a $1 payment, at interest%_per_period and number_of_compound_periods, returns the present value.

SQ Square.

```
«variable» = SQ( «argument» )
```

X^2. Returns the argument squared.

Argument Evaluates To:	SQ Returns:
SQ(2.04939015319)	4.19999999999
SQ(4)	16

SQRT Square root.

```
«variable» = SQRT( «argument» )
```

Returns the square root of the argument.

Argument Evaluates To:	SQRT Returns:
SQRT(4.2)	2.04939015319
SQRT(16)	4

The argument must be positive or the calculator will display the error message "ERROR: SQRT(NEG)" during keyboard evaluations or "SOLUTION NOT FOUND" in Solver.

STAT — Enter Statistics menu. [shift][STAT]

The statistics (number list) menu. See page 57.

STO — Store in a register. [STO]

Copy the calculator line to a register or menu variable. See page 34.

TAN — Tangent. [TAN]

```
«variable» = TAN( «argument» )
```

Returns the tangent of the argument. Uses the current angle mode setting. The inverse of ATAN.

Argument Evaluates To:	TAN Returns:
TAN(86.1859251657)	15
TAN(45)	1

Be sure that your calculator is in Degrees or Radians mode as needed.

TANH(x) — Hyperbolic Tangent. [shift][HYP] TANH

```
«variable» = TANH( «argument» )
```

Hyperbolic Tangent.

TIME — Enter Time menu. [shift][TIME]

The TIME menu performs time calculations, schedules appointments, and sets the clock.

TRN Truncate.

```
«variable» = TRN( «argument» : «number_of_decimal_places» )
```

Returns the argument truncated to number_of_decimal_places. Unlike RND, TRN simply drops the fractional part beyond specified accuracy.

Argument Evaluates To:	TRN Returns:
TRN(4.7196 : 2)	4.71
TRN(−2.531761 : 4)	−2.5317

TVM Enter Financial (TVM) menu. [■][TVM]

The Time Value of Money (financial calculations) menu. See page 62.

USFV Future Value of Uniform Payment.

```
«variable» = USFV( «interest%_per_period» : «number_of_payments» )
```

For a uniform series of $1 number_of_payments and interest%_per_period, returns the future value.

USPV Present Value of Uniform Payment.

```
«variable» = USPV( «interest%_per_period» : «number_of_payments» )
```

For a uniform series of $1 number_of_payments and interest%_per_period, returns the present value.

XCOORD [■][CONVERT] MORE XCORD

```
«variable» = XCOORD(«R» : «angle» )
```

Polar and rectangular coordinate conversions. Uses the current angle mode setting.

XOR Logical exclusive or.

```
«variable» = IF(«conditional_argument» XOR «conditional_argument» :
  «expression» : «expression» )
```

Logical (Boolean) XOR is used with IF. See IF and page 81.

YCOORD

[■][CONVERT] MORE YCORD

```
«variable» = YCOORD(«R» : «angle» )
```

Polar and rectangular coordinate conversions. Uses current angle mode setting.

e^x Exponential.

[■][e^x]

```
«variable» = EXP( «argument» )
```

See EXP.

x^2 Square.

[■][x^2]

```
«variable» = SQ( «argument» )
```

See SQ.

y^x Exponentiation.

[■][y^x]

```
«variable» = «argument» ^ «argument»
```

Raise to a power.

Argument Evaluates To:	Exponentiation Returns:
5^3	125
4^6	4096

$\sqrt{x}$ Square Root.

[■][$\sqrt{x}$]

See SQRT.

10^x ALOG [SHIFT] [10^x]

```
«variable» = ALOG( «argument» )
```

See ALOG.

1/x Reciprocal, 1/x. [1/x]

```
«variable» = INV( «argument» )
```

See INV.

+ Addition. [+]

```
«variable» = «argument» + «argument»
```

Addition.

+/- Change sign. [+/-]

Change the sign of the number in the command line. Negative numbers become positive, positive numbers become negative.

- Subtraction or logical comparison. [-]

```
«variable» = «argument» - «argument»
```

```
«variable» = IF(«conditional_argument» = «conditional_argument» :
  «expression» : «expression» )
```

See page 81 for logical comparisons.

x Multiplication. [x]

```
«variable» = «argument» x «argument»
```

Multiplication.

÷ Division. [÷]

```
«variable» = «argument» + «argument»
```

Division.

% Percent. [■][%]

Percent. Does not work in Solver.

%CHG Enter percent change menu. [■][%CHG]

Perform percent change calculations. See page 53.

^ Exponentiation. [■][y^x]

```
«variable» = «argument» ^ «argument»
```

See y^x.

> Greater than.

```
«variable» = IF(«argument» > «argument» : «expression» : «expression» )
```

Logical (Boolean) greater than comparison used with IF. See page 81.

< Less than.

```
«variable» = IF(«argument» < «argument» : «expression» : «expression» )
```

Logical (Boolean) less than comparison used with IF. See page 81.

= Equal to. [=]

```
«variable» = IF(«argument» = «argument» : «expression» : «expression» )
```

Calculation terminator. Use instead of [INPUT] to evaluate calculations in Stat menu without entering results into the list. Also a logical (Boolean) equal to comparison used with IF. See pages 14 and 81.

π Pi. [■][π]

```
«variable» = «argument» x PI
```

Returns the constant PI (3.14159265359). PI requires no argument.

>= Greater than or equal to.

```
«variable» = IF(«argument» >= «argument» : «expression» : «expression» )
```

Logical (Boolean) greater than or equal to comparison is used with IF. See IF and page 81.

<= Less than or equal to.

```
«variable» = IF(«argument» <= «argument» : «expression» : «expression» )
```

Logical (Boolean) less than or equal to comparison is used with IF. See IF and page 81.

<> Not equal.

```
«variable» = IF(«argument» <> «argument» : «expression» : «expression» )
```

Logical (Boolean) not equal to comparison is used with IF. See IF and page 81.

▲ Rotate history stack up. [▲]

Rotate history stack or stat list up one item. See page 28.

▼ Rotate history stack down. [▼]

Rotate history stack or stat list down one item. See page 28.

[^] Menu key. [^]

See page 17.

← Delete character left. [←]

Delete character to left or cursor. See pages 25 and 46.