Bulletin CB-195A

Texas Instruments Handheld programmable calculators.

The SR-52 A card programmable.

The SR-56 A key programmable.

Plus, the PC-100 printer ...to use with both.

The era of personal programming is here.

Personal programming is here. Economical programmable calculators may well be more significant to business and industry than were slide rule calculators introduced just a short time ago.

Why? Because the programmable calculator introduces a new dimension in problem solving. It decentralizes and personalizes the decision-enhancing power of the computer – bringing to the individual what before was available only to the organization.

Now you can cope with more data, explore with more insight, far more successfully than ever before. At the moment it is most important. Immediately!

So you make better decisions chosen from more options – better decisions founded on a broader data base. Better decisions from more fully optimized trade-offs. Better decisions in a profession where better decisions are the name of the game.

Indeed a programmable calculator is a powerful personal mathematical resource. And you don't need to know computer programming to put it to work. In fact, there's no special language to learn.

Chances are, you already own a calculator – perhaps a sophisticated one. Chances are, too, that you found it exceeded your expectations right from the start – that you grew into it, and it magnified your professional capability far in excess of its cost.

Now personal programming is here. A step-function increase in capability over sophisticated slide-rule calculators. Capability you can put to work now to further strengthen your professional contribution. Capability you won't fully discover until you've owned one and explored its potential for yourself. Capability to enhance decisions of far greater importance than the cost of the model you choose. You will find your programmable is a high-leverage investment.

Optimization. Projections. Forecasting. Data reduction. What-if matrices. Iteration. Risk analysis. Probability. Mathematical modeling. Worst case analysis.

If you're a professional – or studying to be one – then chances are you've got more than a casual interest in these subjects and others like them. In today's tough, competitive, high-technology industrial/business environment these decision enhancing techniques are vital.

Indeed you may already be employing such methods right now, especially if you have your own calculator and have the math skills to do these kinds of analyses. But long calculations and highly analytical math can often be very involved, troublesome, and error-prone – assuming you have the luxury of time to work them through. Otherwise, you get in line to get on the computer. So, more often than perhaps you'd like to admit, you trust to your professional intuition. Make educated guesses. Or do some ballpark figuring.

But you can change all this. No longer do you need to guess. *You can know.* With a programmable calculator from Texas Instruments. Available at prices well within the reach of almost every professional.

You can do it.

You already know how to program – or almost. Whenever you perform a series of calculations, then bring them together to get an answer, you're programming. Except you keep most of it in your head, making each decision as you go.

In fact, you can do a great deal of programming and never use more than the four basic functions (add, subtract, multiply, divide). Programming is natural, you can express your personal approach to problem solving. Whether handling daily currency conversions or solving equations in celestial mechanics.

What does programmability give you?

The ability to tell a calculator: How to perform your calculations your way – repetitively if you wish. How to make detailed, logical decisions for you – based on your inputs or intermediate results. And, how to handle several different subset problems and then combine them to solve your overall problem. *Better decisions by far.*

You'll wonder how you ever got along without a personal programmable.

More than likely you won't be able to write optimum programs straightoff. Programs which run the fastest and use the fewest steps. However, in just a couple of hours, you can begin writing programs that work. Your programmable calculator will probably far exceed your preconceived ideas of its real value. You'll grow into it. As you discover that it can magnify your professional capability, better decisions will be as near as your fingertips.

Just one bad choice can be extremely costly. Just one better decision can pay for your programmable many times over.

You can make decisions anywhere, because a personal programmable calculator goes where you go. In the conference room. In the lab. On the drawing board. Or in the field. With it you have the ability to transform data into action. Right where it's generated. On the spot. Fast. With precision and accuracy.

Optimize your answers.

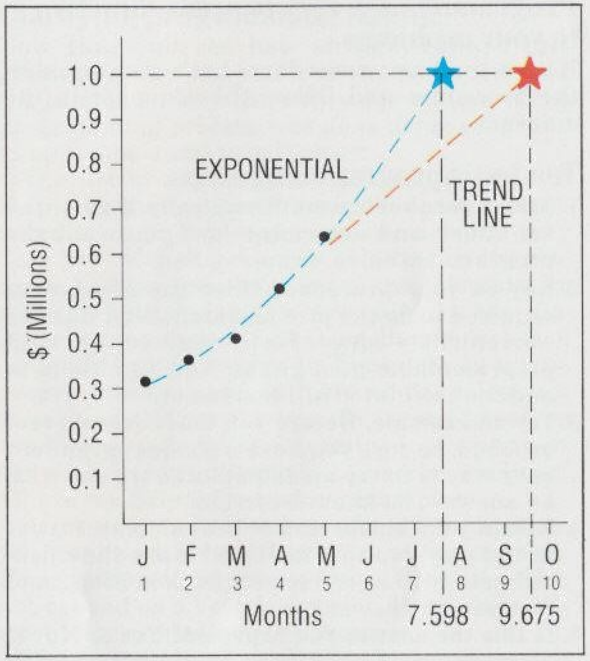
Whether you're in business exploring minimum cost/maximum return on investment. Or in industry, investigating amplifier phase shift, or minimizing noise and vibration. You can ask yourself "what if", then test your assumptions. Or, you can sensitivity test your results against critical parameters. You can explore alternatives. Once a program is developed you can run it again and again. And each time have the program act on different variables entered by you.

Due to the difficulty of photographing calculator readouts, displays represented in this brochure are simulated.

Forecast and estimate more tightly.

Suppose new product sales climb like this:

January	$317,400
February	361,025
March	417,350
April	525,910
May	644,515



You've determined that when sales reach a million dollars a month the product will be profitable. With a TI programmable, your "ball-parking" days are over. A trend line, or a regression analysis, like this one, will tell you the product will make it in mid-September.

But perhaps sales are growing exponentially in the early months. You can also quickly compute a new projection with an exponential curve-fit. This model projects you'll break-even in mid-July. On a programmable calculator you can find these answers in a couple of minutes*.

*See a PC-100 printout of this problem, page 15

Test your data.

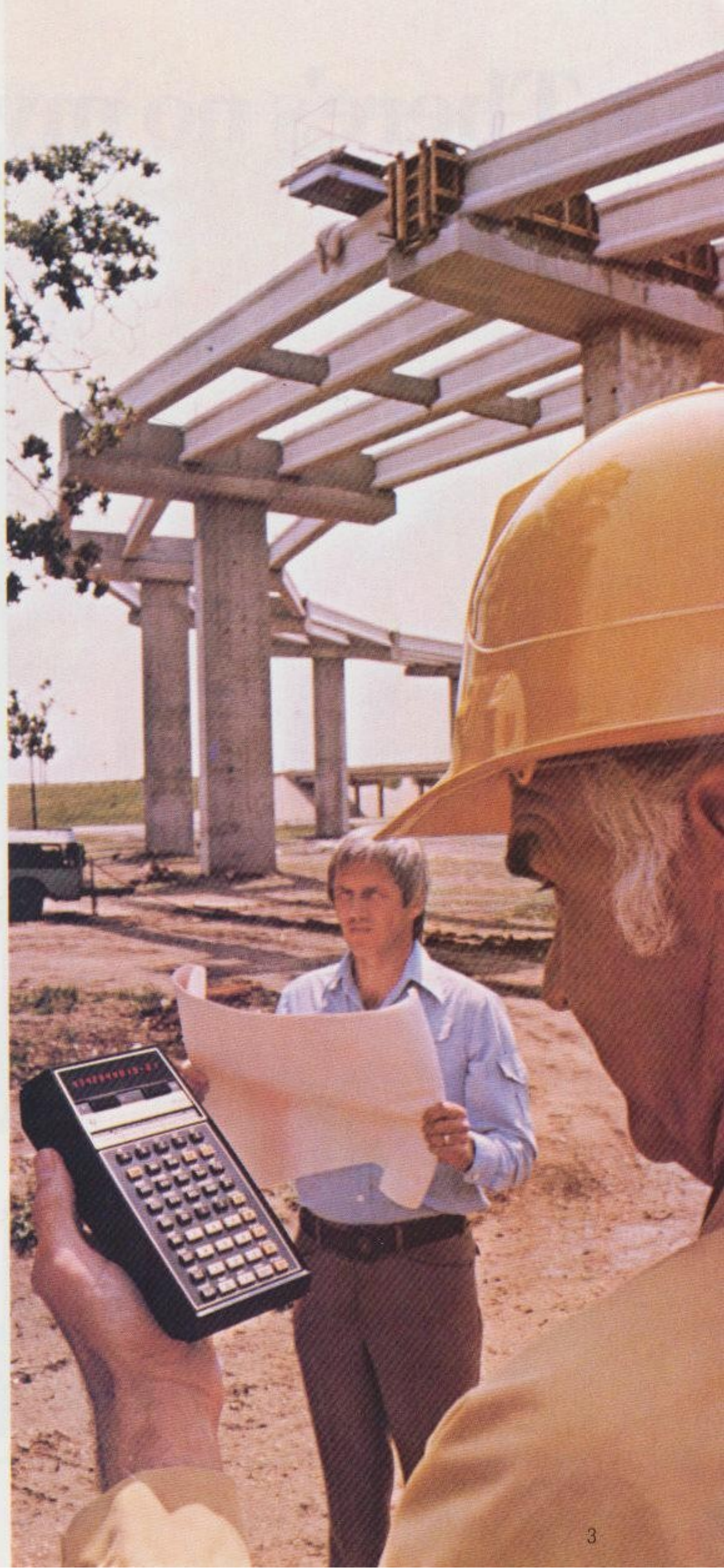
In medicine, social sciences, physics – whatever your discipline. Quickly, and in real time. Examine its quality and significance, and later do a detailed data reduction. The same way it's done on large computers. Only now with far greater personal convenience.

Solve problems iteratively.

A powerful numerical technique. You program the calculator to make successively better and better approximations at the answer you require. It calculates until it comes as close, or as precise, as you want the number to be. It lets you solve problems you can't get at analytically. Like non-linear equations. Or, returns on variable cash flow.

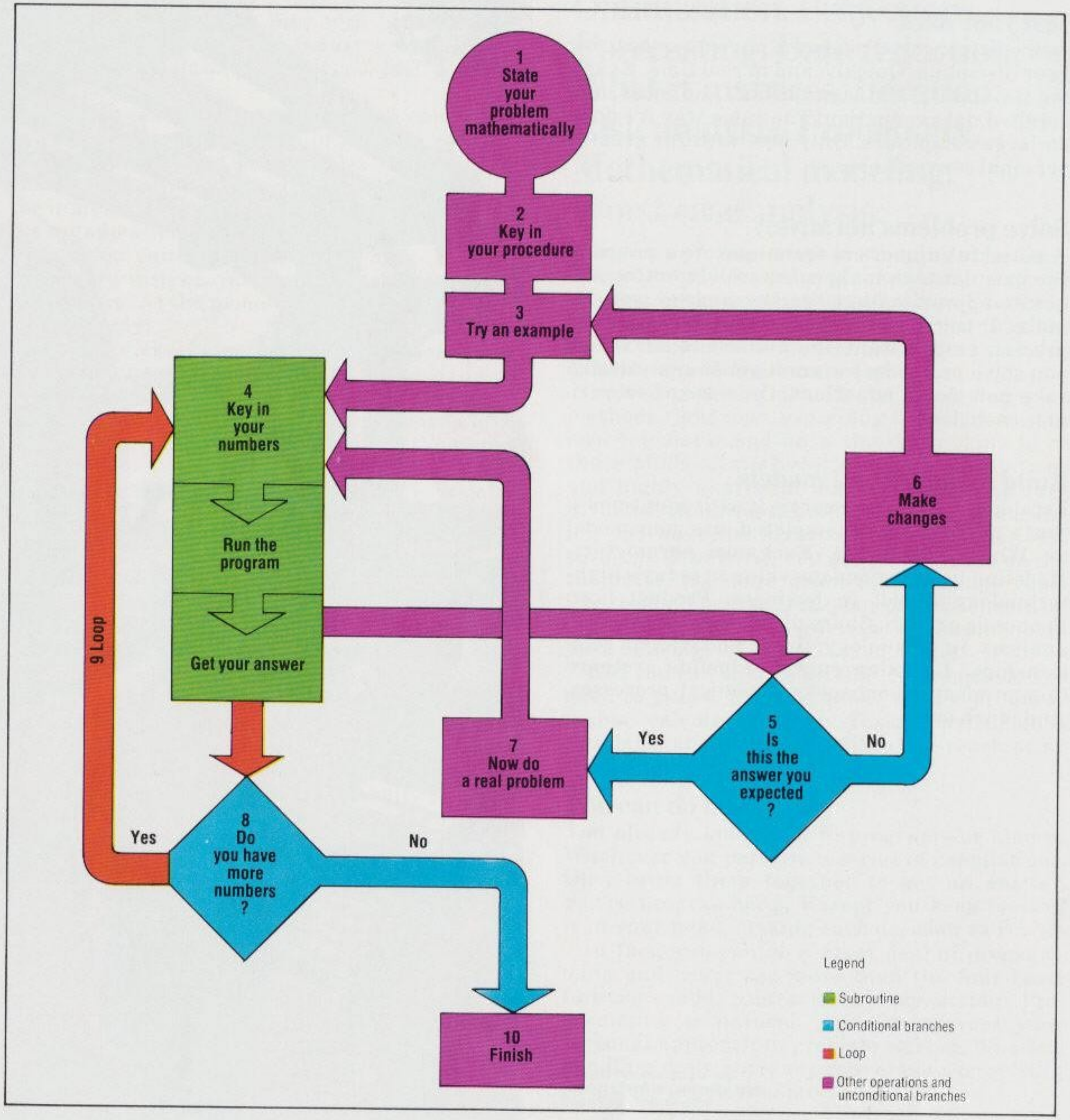
Build mathematical models.

Establish a scenario, express it with equations – that's a model. Once completed, use your model to: What-if. Optimize. Exchange parameters. Modeling has tremendous value in today's high-technology world. In business: Product flow. Economic growth. Share of market. Investment analysis. In technology: Speed and friction relationships. Learning curves. Pipeline systems. Communication networks. Chemical processes. And much more.



Your TI programmable calculator goes where you go. So you can make better decisions on the spot.

There's no mystique to programming.



Programming is just logical thinking. Every problem has a logical flow, from beginning to end. There may be a few constants that must be injected and several variables to be put in which might change the course. Naturally, you have to compensate for these. The same is true in programming.

Programming is easy to apply directly to your problems.

Here's the basic procedure. Let's step through the procedure and follow the sequence in the diagram.

Ten basic programming steps.

1. **State your problem mathematically.** Gather the equations and determine how you want the program to solve them.
2. **Key in your procedure.** List the keystrokes required to do the problem manually. Use the convenient Coding Form that comes with programmable as a guide. Now key them in and the calculator will remember.
3. **Try an example.** Before you start doing a real problem, be sure you have a good program. An easy way is to try an example. So try one with an answer you know is correct.
4. **Key in your numbers.** Let the calculator try it in the way you told it. It will make the calculations which were keyed in back in Step 2 and give you an answer.
5. **Is this the answer you expected?** Yes or No. If No, then you'll want to re-exai ine what you keyed-in and...
6. **Make changes.** Step forward or backward through the program as necessary. Make insertions, deletions, or changes. Now go back and try your example again. This time when you reach Step 5 your answer looks good.
7. **Now do a real problem.** Your program is structui ed and tested – ready for your numbers. No need to key-in the program again, only the variables. The calculator will do the work, and give you the answer.
8. **Do you have more numbers?** Here you can explore options: Ask what-if. Optimize. Sensitivity test your assumptions. Or, determine

what happens under worst-case conditions. Take the Yes path.

9. **Loop.** Here's the real value of a true programmable calculator. Because your work is done. From here on you get answers – all the answers you need. Automatically.
10. **Finish.** With the SR-52 you can permanently record your program on magnetic cards to use again and again. Or, with the optional PC-100 you can print the full contents of your program memory on tape.

Clarifying programming jargon.

Now that you see how straightforward programming can be, you'll begin to understand the jargon. In fact, the entire description of the programming process was done using programming symbols and terminology.

The list of keystrokes which you entered in Step 2 is the *program.* The calculator remembered it in its internal *program memory.* Any time you wished you were able to *run the program.* That is, you commanded the calculator to perform every step you keyed-in. In the diagram you made *decisions* (blue diamonds) based on *conditions* (a good or bad answer). In a true programmable, you can instruct the calculator to make decisions for you. Decisions based on conditions that you've determined: Positive or negative. Zero or otherwise. An error condition. All are *conditional branching* examples.

After you made changes, *edited* the program, you went back for another crack at the example. Here you made an *unconditional branch* – does not depend on a Yes/No decision. Meanwhile, the calculator remembers the changes.

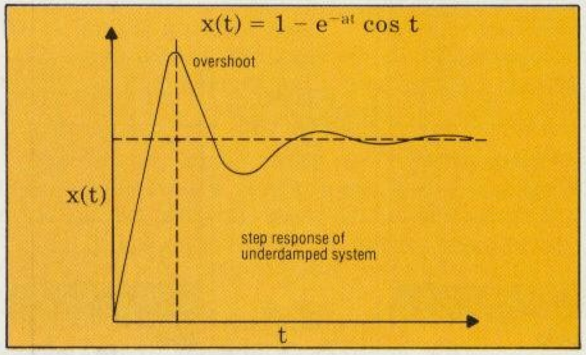
The procedure of keying in the numbers and running the program to get an answer was done in several places. When a procedure can stand alone it is called a *subroutine* (green). Going to a subroutine is a *call.* When completed, it goes back to where the call was made – a *return.*

A *loop* (red) is no more than running through the same series of keystrokes for as many numbers or variables as you need. So you can generate tables, curves or matrices.

As you can see, the jargon describes rather simple concepts. And that's all you need to begin building programs. Your roadmap is the *flow chart* – the logical, step-by-step, graphical representation of your problem/solution.

Let's try a real problem.

Here's one that occurs in analysis of a digital or analog electronic circuit. Structural vibration. A servomechanism. A shock absorber. Even a sociological or economic model: Determine the overshoot of the step response of an underdamped second-order system as a function of the damping factor, a.



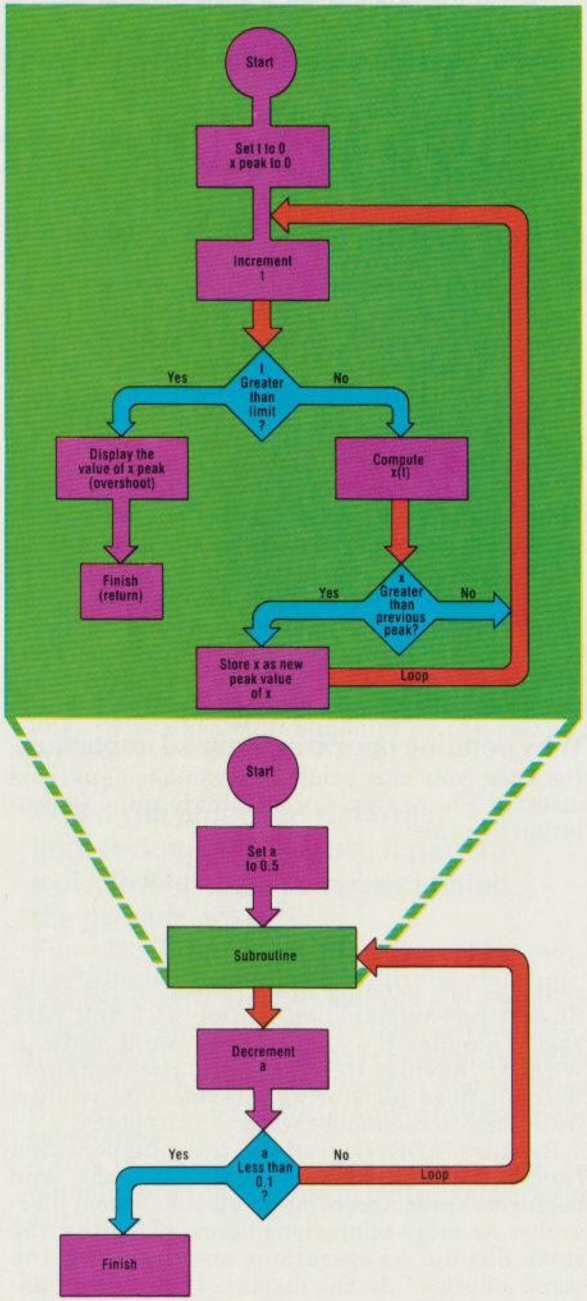
This could be solved analytically. But with an SR-52 or SR-56 you can quickly get numerical answers, which is what you need.

The approach is straightforward.

Just program the calculator to compute the value of x (t) for a specific t, and compare that value to the previously determined peak value. If the answer is less than the previous peak, go on and compute x (t) for the next value of t. If this answer is greater than the previous peak, then make it the new peak value and continue for the next value of t, until maximum t is reached (top diagram at right).

Now we want this procedure repeated for several values of "a": 0.5 to 0.1 in increments of 0.1. In the SR-52 and SR-56, we can define the entire procedure as a *subroutine* and write a new, smaller program which uses this subroutine procedure in a loop. The top subroutine can be added to the end of this small program – and be *called* from it (bottom diagram).

Notice how programs are built-up. From simple blocks into increasingly complex programs.



TI's unique Algebraic Operating System distinguishes the SR-52 and SR-56 programmables.

With the introduction of the SR-50 slide rule calculator a few years ago, Texas Instruments had a choice: algebraic entry or Reverse Polish Notation (RPN). TI chose algebraic entry because it's the most natural and easiest to use.

Now, with the new SR-52 and SR-56 programmable calculators, TI takes another major step forward in power and ease of use—the unique Algebraic Operating System.

What is AOS?

Actually, it's easier to use than to explain. AOS is more than just algebraic entry. It's a *full* mathematic hierarchy coupled with multiple levels of parentheses. This means more pending operations, as well as easy left-to-right entry of expressions—numbers and functions.

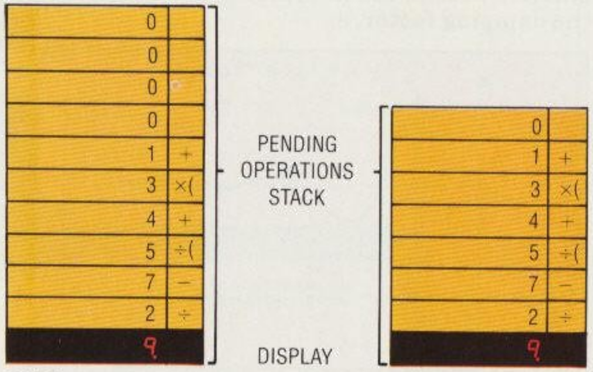
Why pending operations are so important.

Because you can compute complex equations directly. For example, a seemingly simple calculation like this:

$$1 + 3 \times \left[4 + \frac{5}{(7 - \frac{2}{9})} \right] = ?$$

contains six pending operations. An SR-52 or SR-56 programmable calculator with full AOS easily handles it—just as you would state it. Without having to rearrange the equation. Without need for storing intermediate results. Or having to remember what's in the stack.

Because AOS *remembers* the numbers and functions in its pending operations stack. And performs them according to mathematical hierarchy. As more operations become pending, the stack fills up. As operations are completed, the stack empties into the display. Here's what the stack contains when you key in the calculation:



SR-52
9 levels of parentheses
10 pending operations
11-register stack, including the display

SR-56
9 levels of parentheses
7 pending operations
8-register stack, including the display

Mathematical hierarchy.

This is the universally recognized order of performing calculations. Functions first. Powers and roots. Multiplication or division. Then addition or subtraction. AOS performs calculations in this order. But you have the option to change the order whenever you wish by using the parenthesis keys. We did that in the above example. With AOS, when you push the equal key, the answer (15.21311475) is computed using the correct order of execution.

AOS makes the calculator part of the solution. Not part of the problem.

The case for AOS is strong. That's why TI chose it. Whether you own a calculator with ordinary algebraic entry, or RPN or no calculator at all, we think you'll prefer AOS. Because you begin using it immediately. There's no new language to learn. Even if you are conditioned to RPN, the added value and power to TI's programmable calculators with unique AOS is well worth the easy transition.

TI's advanced memory technology provides extraordinary capability.

TI is in the forefront of semiconductor memory technology. And memory is vital when it comes to building powerful programmable calculators.

More memory means more internal working registers.

The pending operation stack which makes AOS so powerful, contains 11 registers in the SR-52 and eight registers in the SR-56. Compared to four registers in leading RPN calculators. Also, the AOS stack remembers the operations and the numbers, whereas RPN calculators store numbers only—you have to remember the operations.

More memory means more program steps.

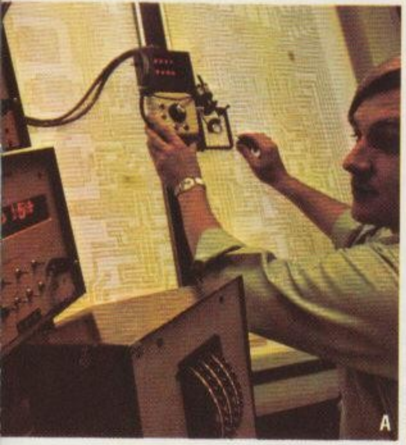
The SR-52 and the SR-56 give you more program steps than comparable calculators and that means you can write programs which solve more lengthy problems. Programs don't have to be shoehorned into an inadequate program memory. You don't have to juggle. Just key-in your program and problem solving by programming is an easy, practical thing to do.

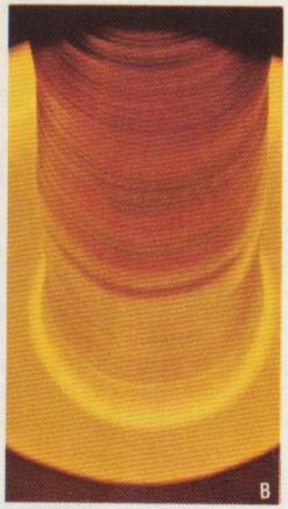
More memory means more data registers.

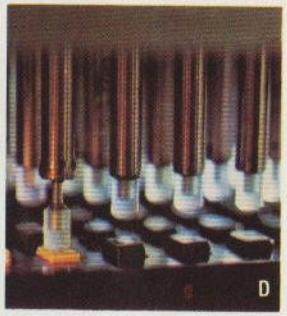
With 20 data registers in the SR-52, you can store more variables. Because AOS stores numbers and pending operations, the user-accessible data registers are at your complete disposal—for arrays, input data, multiple outputs. The SR-56 has 10 data registers.

More user-accessible data registers means you can look at the whole problem in one program. No need to break it up into several programs. You can consider *all* the factors which affect the results. So you can perform what-if and optimization right in the program. It also means you can solve more complex problems. For example, you can find the internal rate of return for a ten-period variable cash flow stream. Or, solve a three-by-three matrix. (Both are in the SR-52's optional library.)

TI's goal: Greater value. Higher performance. Lower price. Quality. Without compromise.







A. **Technological leadership.** TI designs in quality, every step of the way.

B. **Crystal puller.** TI quality begins at the material level with high-purity silicon.

C. **Probe test.** Computer-controlled testing checks basic parameters of each integrated circuit.

D. **Pneumatic tester.** Final testing examines key operation and calculator functions.

How does TI – time and again – provide advanced products, like calculators, with greater capability than our most respected competitors at a substantially lower price?

You can bet it's not by quality shortcuts. We've got too much to lose.

The engineer or manager who buys our calculator may also specify millions of dollars worth of TI microprocessors or corporate-wide computer systems.

The key to better value in TI calculators is technology. Plus, a very tough-minded management philosophy that says, design to cost objective. Reduce prices as costs permit. But never compromise our reputation with shortcuts in quality.

Instead, TI uses the tremendous leverage of technological know-how. Coupled with an almost total in-house start-to-finish capability. This is what brings you unchallenged value.

In high-technology products, higher price does not always mean higher quality.

The key is the ability of advanced microcircuit technology to combine ever greater electronic capability in fewer and fewer components. It results in substantial savings throughout. Fewer components to design in . . . to purchase . . . to inventory . . . to inspect . . . to solder . . . to test. It means elimination of associated parts . . . circuitry . . . PC boards . . . sockets. As well as all *their* related inventory . . . handling, testing, scrap costs, and more.

Fewer components mean fewer parts to fail, fewer interconnection faults. Reliability increases.

What's more, the higher the volume and the greater the experience with these high-technology components, the better process control. The higher the yields. And the lower cost per unit.

The combination of technological know-how and volume production experience results in substantial savings that TI passes along by way of lower prices. The same process of parts minimization pays off handsomely in increased reliability for the user. The result is that prices come down as performance goes up. While quality improves as more is learned about the technologies.

No other company in the world has a stronger claim to that know-how than TI.

But TI value doesn't stop there. We're steeped in calculator technology from start to finish. We make the keyboards. The circuit boards. The displays and all associated solid-state components – practically everything but the batteries.

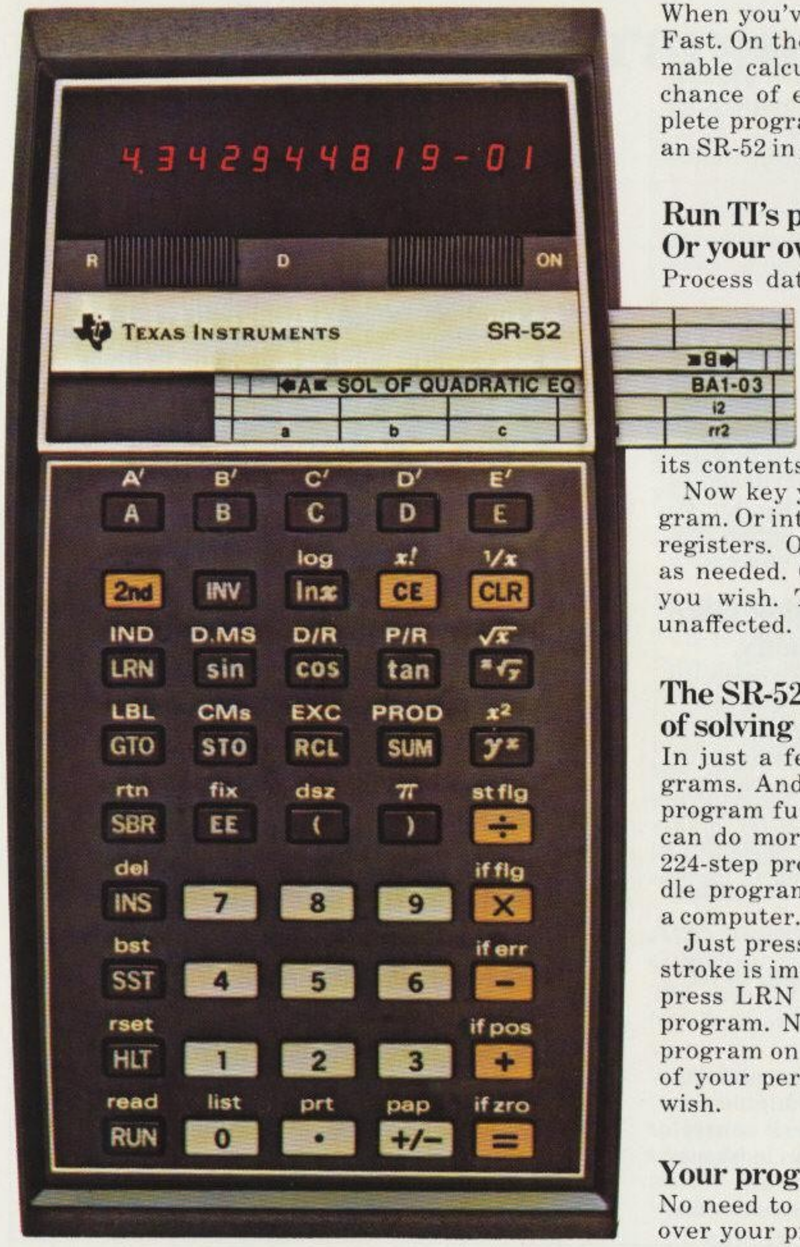
The same tough design-to-cost philosophy permeates every division's contribution. Stringent guidelines are applied to cost, quality and performance at every step. Common techniques of quality control – managed by on-the-spot calculator experts – helps insure that no weak links jeopardize the reliability of the complete system.

When you judge the value of a high-technology product, it pays to look closely at the company behind the product.

Know-how is the name of the game. TI invented the original integrated circuit and the "calculator-on-a-chip" that ignited the calculator revolution . . . and is the world's leading producer of integrated circuits. TI holds the basic patent on the miniature calculator itself . . . and is a world leader in the production of electronic calculators.

It is that kind of experience, know-how, and commitment that results in better value. Never compromise.

The SR-52. Adds a vital, powerful new dimension to your work. Wherever it may be.



(Actual size)

When you've got a problem, you want answers. Fast. On the spot. With the new SR-52 programmable calculator you get them. And with less chance of error. Because you can load a complete program from a small magnetic card into an SR-52 in just two seconds.

Run TI's prerecorded programs. Or your own.

Process data or perform complex calculations automatically. Select a prerecorded program from one of the SR-52's optional libraries or from the Basic Library. Or run one that you have written. Load the prerecorded magnetic card and you put its contents into the SR-52's program memory.

Now key your variables directly into the program. Or into one or more of the 20 data memory registers. Or both. Run the program as often as needed. Change values of your variables if you wish. The program stored in memory is unaffected.

The SR-52 can learn your way of solving problems.

In just a few hours, you could be writing programs. And because of AOS, and the powerful program functions of the SR-52, your programs can do more than you could imagine. With its 224-step program memory, the SR-52 will handle programs you may have thought required a computer.

Just press the learn key (LRN) and each keystroke is immediately stored. When you're done, press LRN again. The SR-52 has learned your program. Now it's ready to RUN. Record your program on a blank magnetic card, and it's part of your personal library. Ready whenever you wish.

Your programs are doubly protected.

No need to worry about accidentally recording over your programs. TI has developed a unique optical record protection feature. And, a multi-key recording command virtually eliminates that possibility.

Its flexible branching capability is like a computer's.

Program steps are usually processed sequentially. But sometime you'll want to handle them out of order. The SR-52 has two ways to branch: Unconditional. Conditional.

Three types of unconditional branches. Unconditional branching tells the SR-52 to skip to another part of the program. You may program three types: Go to. Subroutine. Reset. These lead to locations you specify by: An absolute program memory location. A label. Or, a variable address specified in data memory.

Unconditional Branches	To An Absolute Prog. Memory Location	To A Label	To An Address* Specified In A Data Memory
Go To	Yes	72	20
Subroutine	Yes	72	20
Reset	Yes	–	–

*Indirect Addressing

Branch to an absolute location. This is the quickest, most direct way to branch. Tell the SR-52 the number of the program memory location you want to branch to—and it's done.

Branch to a label. Labels let you identify a specific program memory location by a symbolic name. Virtually every key can become a label. Including second functions. You can save keystrokes. Simplify editing. Tighten up the entire program.

Branch to an address specified by a data memory. With absolute addressing you tell the calculator which program memory location to branch to. With indirect addressing you tell the calculator to branch to a location specified by a data memory. This lets you get to different places at different times in a program. It can also increase your number of subroutine levels.

Branch to user-defined keys. These keys may become any function you want them to be. There are two ways to use them: From the keyboard to enter data or perform complex calculations. Or in a program, where they become automatic subroutines.

Ten types of conditional branches. These are decisions made by the SR-52. They depend on tests. If test conditions are met, then a branch takes place. Otherwise, the normal sequence continues.

There are six display tests (positive, negative, zero, non-zero, flashing, not flashing). Two flag tests (flag set or cleared). And, two looping tests based on the contents of data memory 00 (zero or non-zero).

As with unconditional branches, you can make conditional branches to locations you specify by: absolute addresses, labels, or indirect addresses.

Conditional Branches		To An Absolute Prog. Memory Location	To A Label	To An Address* Specified In A Data Memory
If Positive	Direct	Yes	Yes	Yes
	Inverse	Yes	Yes	Yes
If Zero	Direct	Yes	Yes	Yes
	Inverse	Yes	Yes	Yes
If Error	Direct	Yes	Yes	Yes
	Inverse	Yes	Yes	Yes
If Flag	Direct	5	5	5
	Inverse	5	5	5
DSZ (Loop)	Direct	Yes	Yes	Yes
	Inverse	Yes	Yes	Yes

*Indirect Addressing

Five flags. These are on/off signals, or switches. Each is set or cleared by you. From the keyboard or in a program.

Two loops. Suppose you wanted to run a calculation 144 times. Store 144 in data register 00. Key in DSZ at the end of your calculation, and sit back and watch. The SR-52 does it – 144 times.

Twenty data memories you can access directly or indirectly.

Direct register addressing means you can store a number in a memory register you designate. Five STO 10, for example, means store 5 in data memory 10.

Indirect addressing allows you to store the number in a data memory specified by another register. Suppose your indirect address instruction is: Five, indirect store 10. This means store five. Not in register 10. But in the register whose *address* is found in register 10.

Indirect addressing is a versatile tool for: Handling arrays of numbers. Matrix operations. Or classifying data in cells, as in histogram construction. With an SR-52 you can: Store. Recall. Exchange. Add. Subtract. Multiply. Divide. Directly or indirectly.

Edit and debug your program.

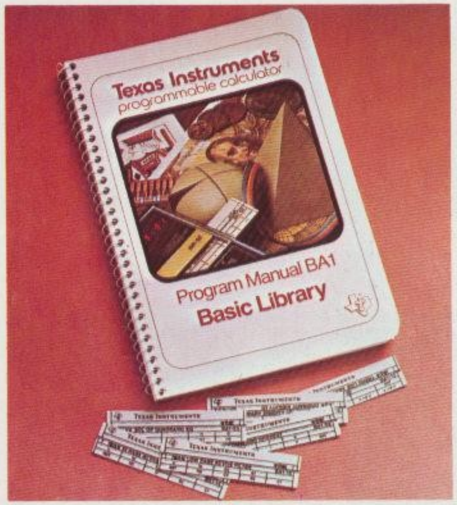
Move through a program a step at a time. Forward, or backward. Inserting (adding more steps) moves your current and all following instructions down one location. Delete moves the following instructions up. Or write over steps. A convenient single-step key lets you go through a program one step at a time when editing. Or use it to run the program a step at a time.

Insert the prerecorded magnetic card. (A side). Remove card. Send it through again. (B side).

Complete a program, record it. Permanently. Now it's in memory and on the card.

Also a powerful slide rule calculator.

The SR-52's value and power comes through even when you don't program: Trig functions and inverses. Logs and exponents. Powers and roots. Degrees/minutes/seconds to decimal conversion. Polar/rectangular conversion. Degree/radian conversions. Factorials. Reciprocals.



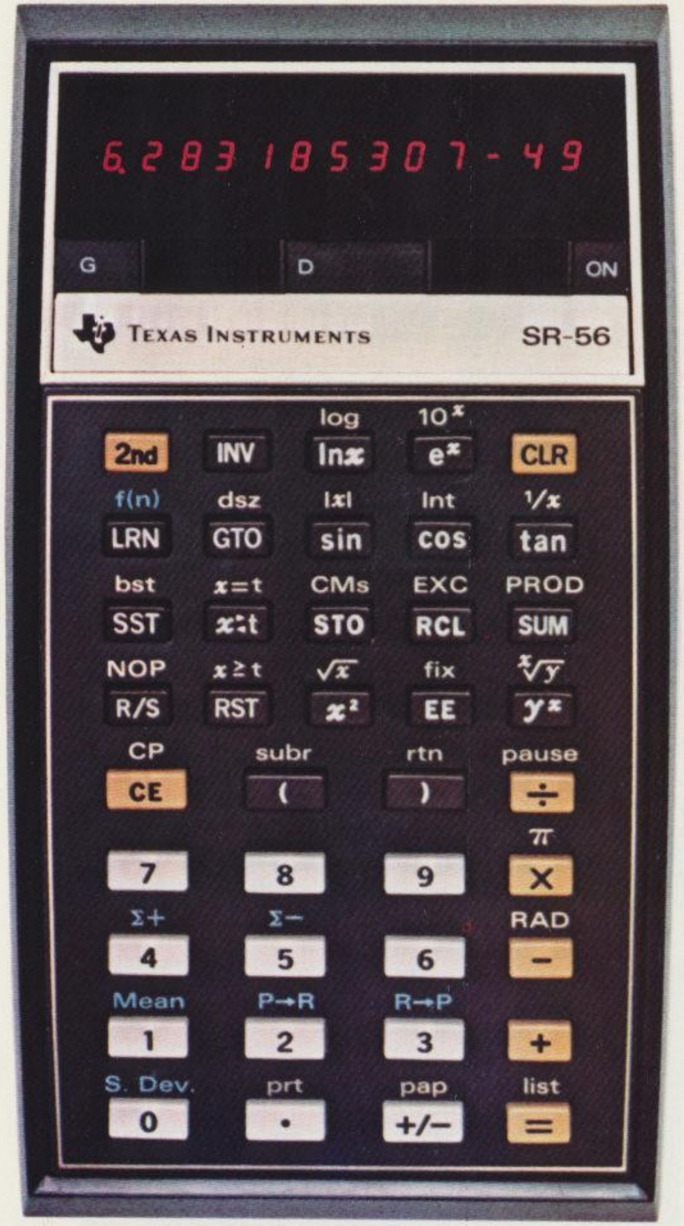
SR-52 Basic Library and prerecorded programs.

Twenty-two prerecorded program cards come with an SR-52. You can put them to work right away. You also get a 96-page Basic Library manual. Each prerecorded program card is supported with sample problems, user instructions and program listings. See optional libraries on pages 12-13.

• Conversions (1,2) • Solution of Quadratic Equation • Hyperbolic Functions • Prime Factors of an Integer • Complex Arithmetic • Checkbook Balancing • Compound Interest • Ordinary Annuity (1,2) • Trend Line Analysis • Permutations and Combinations • Statistical Means and Moments (1,2) • Random Number Generator • High Pass Active Filter • Low Pass Active Filter • Dead Reckoning • Lunar Landing Game • Diagnostics

A head cleaning card and a generous supply of blank magnetic cards are also provided.

The SR-56. Programmability for better decision-making. Tremendous math power. Outstanding value at an economical price.



(Actual size)

Whatever the application, the SR-56 easily becomes an integral part of your work. As a powerful slide rule calculator. Or as a versatile key programmable. It's capable of solving problems once handled by computers.

Its power is amplified with TI's new unique Algebraic Operating System, coupled with extensive programming capability. Every feature is designed to smoothly handle your problems with its: 100-programming steps. An eight register stack that handles up to seven pending operations. Plus, nine levels of parentheses. And 10 data memories.

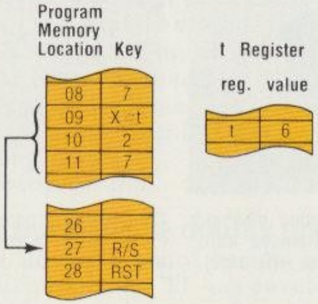
Branches like a computer.

Just as with the SR-52, program steps are usually processed as they're entered. But sometimes program steps need to be handled out of sequential order. This change is called branching. The SR-56 is capable of direct addressing which means you simply specify the instruction you wish it to go to.

The SR-56 has three unconditional branches which include: Go to. Reset. Subroutine (4 levels). And six conditional branches, which include two for loop control and four test register comparisons.

A unique independent test register.

Lets you compare the value in the display with the value in the test register—without interfering with the processes in progress. If your test conditions are met, then a conditional branch is made. Otherwise the sequence continues. This sketch shows a conditional branch to instruction 27. If the t-register had contained an eight, then no branch would be made. The program would go on to location 12.



Program Memory Location	Key
08	7
09	X:t
10	2
11	7
26	
27	R/S
28	RST

t Register

reg.	value
t	6

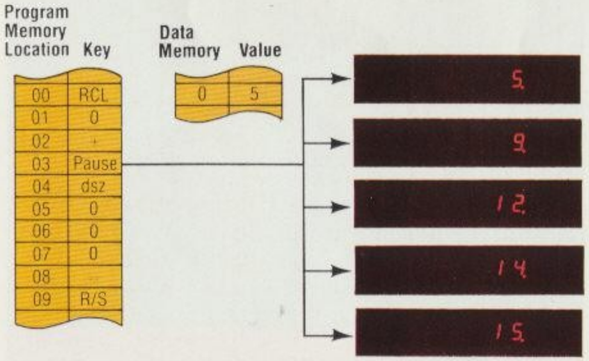
There are four types of tests which you can monitor at the display: Greater than or equal. Less than. Equal. Not equal.

Enough memories to do your tough problems.

With 10 user memories, you can store and recall data. Add, subtract, multiply or divide directly within a memory register. Without affecting the calculation in progress.

A unique pause key that works two ways.

Let's illustrate. Here's a program that sums consecutive integers: $\sum_{0}^{N} k$. RCL 0 recalls the integer k from data memory 0 (5, in this case). This is summed (+) and the result displayed by the pause instruction in location 03. Decrement-and-skip-on-zero (DSZ) instructs the SR-56 to reduce the integer by one, loop back to location 00 and repeat the summation with the next integer. The loop continues until data memory 0 has been decremented to zero which completes the program. Meanwhile, the display shows each step in the summation: 5 + 4 + 3 + 2 + 1.



Program Memory Location	Key
00	RCL
01	0
02	+
03	Pause
04	dsz
05	0
06	0
07	0
08	=
09	R/S

Data Memory	Value
0	5

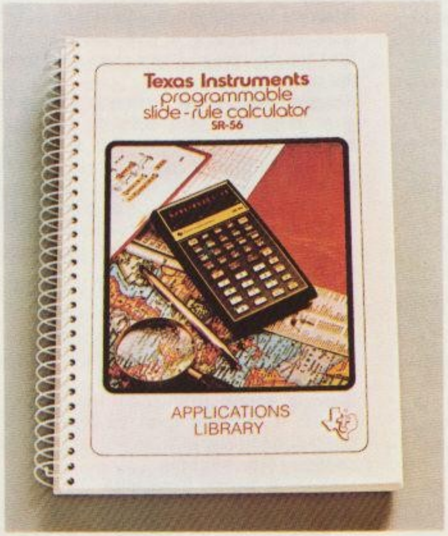
Or, hold the pause key down and you'll see the result of every step in your program (½-second a step).

Editing is easy.

Single-step and back-step keys let you quickly sequence through program memory to detect errors or examine what you've done. If you push a key you didn't mean to, you can write-over it with the NOP key.

And it's a powerful slide rule.

With 74 preprogrammed functions and operations, the SR-56 can handle your math problems quickly. From logarithms and trigonometry (degrees, radians or grads), to more advanced statistical problems, and polar/rectangular conversions.



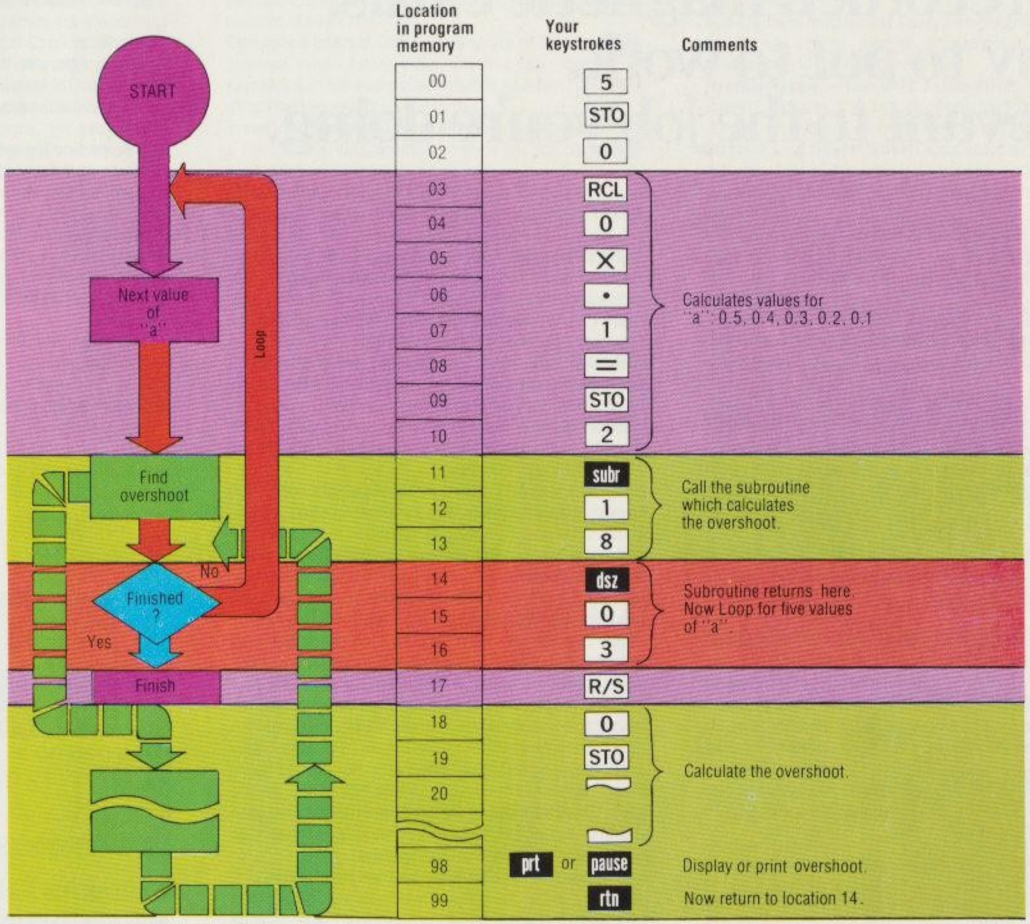
An applications library, too.

A 192-page collection of programs. All prewritten. Select a program. Follow the listing (putting in your own data, of course). And, you'll immediately begin using your SR-56's computing power to solve your own problems.

Every program in the Applications Library was chosen specifically on the basis of occupational demands. Each program contains a thorough description of how it works and the conditions under which it operates. There are also extensive examples of each program in typical problem solving situations.

• Math • Statistical • Financial • Electrical Engineering • Navigation • others

Solve a problem on the SR-56 and print answers on the PC-100.



Location in program memory	Your keystrokes	Comments
00	5	
01	STO	
02	0	
03	RCL	Calculates values for "a": 0.5, 0.4, 0.3, 0.2, 0.1
04	0	
05	X	
06	•	
07	1	
08	=	
09	STO	
10	2	
11	subr	Call the subroutine which calculates the overshoot.
12	1	
13	8	
14	dsz	Subroutine returns here. Now Loop for five values of "a".
15	0	
16	3	
17	R/S	
18	0	Calculate the overshoot.
19	STO	
20		
98	prt or pause	Display or print overshoot.
99	rtn	Now return to location 14.

Here is the overshoot problem from page 5*.

Let's start at the top, program memory location 00. First we're going to calculate the overshoot for an "a" of 0.5. We enter the integer 5 and store it in data memory 0. At location 03 we recall (RCL) the integer from data memory 0 and multiply it by 0.1 to get 0.5. Then store the result in data memory 2 (STO 2) for later use. Next we call the subroutine (SUBR) in location 18. This is the procedure (not shown in full) for finding the overshoot. When it's finished, it will display (or print on the PC-100) the overshoot, then return to the location following the subroutine call (location 14). Here the calculator decrements (or reduces) the integer value in memory 0 (5) by one. Then it loops back to location 03 again. Looping continues until a zero appears in data memory 0. The calculator then skips to program memory location 17 and stops (R/S). Thus calculating and printing overshoot for five values of "a".

*This diagram has been redrawn to show the relationship between the flow chart and the keystrokes.

Optional libraries for the SR-52. Prerecorded magnetic cards. Easy to put to work. Relevant to the job you're doing.

Optional libraries for the SR-52 go further and do more. Because of the SR-52's 10 user-defined keys, 20 data memories and 224 program steps. This means more steps and functions can be included on a card.

Programs are selected on the basis of usefulness. Each one is thoroughly researched and tested.

An easy-to-follow manual is also included with each Library. Every program is described in detail. Pertinent equations, sample solutions, step-by-step user-instructions are backed-up by a complete listing of the program.

Electrical Engineering Library

Low Pass Active Filter Determines component values for low-pass active filter.

High Pass Active Filter Determines component values for high-pass active filter.

Active Bandpass Filter Design Second-order active bandpass filters are designed using a multiple-feedback network. Both high-Q and low-Q circuits may be realized. Standard values may be selected to minimize implementation problems.

Passive Bandpass Filter Design (1) T and π passive bandpass filters are designed given the pass band and termination impedance.

Passive Bandpass Filter Design (2) Permits plotting the frequency response of a bandpass filter.

Chebyshev Filter Design Chebyshev low-pass filters are designed for specified filter order, termination resistance, corner frequency and allowable ripple.

Butterworth Filter Design Butterworth low-pass filters are designed for specified filter order, termination resistance and corner frequency.

Series Resonant Circuit Impedance and resonant frequency are calculated for a series resonant circuit whose component values are specified. Included is a routine to be used for plotting the impedance over a range of frequencies.

Parallel Resonant Circuit Impedance and resonant frequency are calculated for a parallel resonant circuit whose component values are specified. Included is a routine to be used for plotting the impedance over a range of frequencies.

T and π Attenuators Component values for T and π impedance matching circuits are found for specified input and output impedances and desired loss. Minimum loss pad matching may be performed for given impedances.

T to π Transformation T networks are transformed to π networks having the same characteristics. Impedances for each part of the network are specified.

π to T Transformation π networks are transformed to T networks having the same characteristics. Impedances for each part of the network are specified.

Ladder Network Analysis Input impedance for a ladder network is calculated. The network may be composed of any combination of resistors, capacitors, and inductors in shunt or series addition.

Transmission Line Impedance Impedance of types of transmission lines may be calculated. The types are coaxial, 2-wire, and single conductor near ground.

Transmission Line Impedance Transformation Electrical length of a transmission line is calculated for specified physical length, frequency, and relative permittivity of the dielectric. Input impedance can also be calculated for a specified characteristic impedance.

S and Y Parameter Transformation (1,2) A set of S (Y) parameters expressed as magnitudes and angles is transformed to a set of Y (S) parameters.

Phase Locked Loops Natural frequency, damping factor and loop noise bandwidth are found for either passive or active phaselocked loops. Loop gain and component values for the circuits are required as inputs.

Low Frequency Transistor Amplifier Design Forward current amplification factor, quiescent collector current and biasing resistor values are calculated for specified circuit and transistor parameters.

Discrete Fourier Series Fourier sine or cosine coefficients are computed for discrete values of a periodic function.

Power Transformer Design For specified inputs, core area, output power, frequency, primary turns, input voltage, and flux density are calculated for a power transformer.

Coil Properties The inductance and number of turns of a single or multilayer coil are calculated from the inside radius of the coil, distance between the windings and the winding height.

Controlled Rectifier Circuits For specified phase angle or average output voltage and supply voltage, the supply voltage, phase angle, average output voltage, peak-to-peak output voltage and rms output voltage are calculated.

Power Supply Rectifier Circuits Full-wave or half-wave rectifier circuits are evaluated for given component values, input voltage and frequency. The average dc voltage and peak-to-peak ripple are calculated.

Statistics Library

Basic Statistics for One or Two Variables Calculates the means, standard deviations, and standard errors for one or two variables using both the normalized and unnormalized methods. Also calculates the covariance and correlation coefficient for the two variables.

Permutations and Combinations Calculates the permutation and combination for a given n and r.

Means and Moments 1,2. The arithmetic, geometric, harmonic, and generalized means, the first four moments, and the kurtosis and skewness of distribution are calculated for grouped or ungrouped data.

Random Number Generator Uniformly distributed and normally distributed random numbers are generated.

One Way Analysis of Variance Performs a one way analysis of variance on k treatment groups. Includes the F statistic degrees of freedom and sums of squares.

t Statistic Evaluation (Paired Observations) Uses t statistics to test the difference between the means of two normally distributed populations.

t Statistic Evaluation (Two Sample Test) Evaluates the t statistic with n + n − 2 degrees of freedom.

Linear Regression The slope, intercept, estimated values for x or y, and coefficient of determination for the linear least-squares fit of given points (x,y) are calculated.

Power Curve Fit The slope, intercept, estimated values for x or y, and correlation coefficient for a power least-squares fit of given points (x,y) are calculated.

Exponential Curve Fit The slope, intercept, estimated values for x or y, and correlation coefficient for an exponential least-squares fit of given points (x,y) are calculated.

Logarithmic Curve Fit The slope, intercept, estimated values for x or y, and correlation coefficient for a logarithmic least-squares fit of given points (x,y) are calculated.

Multiple Linear Regression 1,2. The coefficients of the linear least-squares fit of given points (x,y) are calculated as well as the z corresponding to given x and y for this line.

Quadratic Regression 1,2. The coefficients of the quadratic least-squares fit of given points (x,y) are calculated and the value of y for a given x is determined for this curve.

Histogram Construction Constructs a histogram over a given range and given observed data points. The frequencies are calculated for each of 12 cells and the mean and variance are calculated for the entire range.

Normal Distribution The standard normal distribution for a given x is calculated using a probability function.

Chi-Square Distribution The chi-square density function and probability B(½) are calculated, given the degrees of freedom and x.

t Distribution The integral of the t distribution is calculated, given the degrees of freedom and x.

F Distribution The probability of an event X > x (area of the "upper tail") for the F distribution is calculated for two given degrees of freedom and x.

Bivariate Normal Distribution The bivariate density function is calculated given the means of x and y, standard deviations of x and y, and the correlation coefficient between x and y.

Logarithmic Normal Distribution The median, mode, mean, and variance are calculated for the logarithmic normal distribution given m and σ^2. Given an x, f(x) is then calculated for this distribution.

Weibull Distribution The probability density and cumulative probability functions are calculated for the Weibull distribution of given m, n, and x. Given Q(x), the program also calculates x.

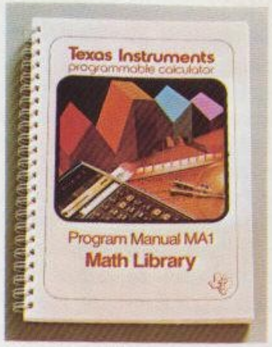
Poisson Distribution Calculates the density function and cumulative distribution of the Poisson distribution given m and n.

Binomial Distribution The binomial density function, the cumulative distribution, mean, variance, and probability of at least k successes are calculated given n, p, and k.

Negative Binomial Distribution The mean, variance, negative binomial density function, cumulative distribution and probability of k or less successes are calculated given r, p, and k.

Geometric Distribution The variance, probability, cumulative distribution, and geometric density function are calculated given μ and x.

Hypergeometric Distribution The hypergeometric density function, cumulative distribution, mean, and variance are calculated given a, b, n, and x.



Math Library

Hyperbolic Functions Calculates all six hyperbolic functions and their inverse functions.

Solution of Quadratic Equation Solves for real and complex root of basic quadratic equation.

Solution of Cubic Equation Solves for a real root of a cubic equation and stores the coefficients for use with MA1-02 to solve remaining real and imaginary roots.

Zeros of Functions Finds the roots of a user defined function using the bisection method. This method finds one root per interval in which the function changes sign.

Simultaneous Equations in 2 or 3 Unknowns Solves a set of simultaneous equations of either two or three unknowns.

LaGrange Polynomial Interpolation Solves for new values of a function f(x) when two to six pairs of x and f(x) are known.

Gaussian Integration (X_0 to X_f) Finds the integral of a user defined function by the six-point Gaussian integration method over the interval (a,b).

Gaussian Integration (X_0 to X_∞) Finds the integral of a user defined function by the six-point Gaussian integration method from a to infinity.

Trapezoidal Integration, Given F(X) Calculates the trapezoidal approximation of a function, where f(x) is defined by the user over a given interval.

Trapezoidal Integration, Given X_N, $F(X_N)$ Calculates a trapezoidal approximation of the area under a piecewise linear curve with known points X_n, $f(x_n)$.

Simpson's Approximation Approximates the area under a curve using Simpson's Rule.

First Order Differential Equations Solves first order differential equations using a numerical third order Runge-Kutta approximation.

Matrix Inversion and Determinant (2 x 2) Finds the inverse matrix and determinant of a 2 by 2 matrix. Also solves for the product of two 2 by 2 matrices.

Matrix Inversion and Determinant 1,2 (3 x 3) Finds the inverse matrix and determinant of a 3 by 3 matrix.

Matrix Arithmetic (1) Performs addition or subtraction upon two m by n matrices where m and n may be 1 through 4.

Matrix Arithmetic (2) Performs multiplication upon a m by n matrix and a n by p matrix where m, n and p may be 1 through 3.

General Matrix Product Performs multiplication upon a m by n and n by p matrix where n may be 1 through 18.

Vector Operations Given two 3-dimensional vectors, finds the magnitude of each vector, the cross product and the dot product.

Partial Sums and Products Computes the partial sum or partial product of a user defined function.

Base Conversions Converts real numbers from decimal to any base from base 2 through 99 or vice versa.

Prime Factors of an Integer Determines all prime factors of an integer.

Greatest Common Divisor and Least Common Multiple Given two integers, finds the greatest common divisor (GCD) and the least common multiple (LCM).

Arithmetic and Harmonic Progressions Finds the arithmetic and harmonic progressions, the sum of the first n terms of an arithmetic progression and the n^{th} term of an arithmetic or harmonic progression.

Geometric Progression Finds the geo-metric progression, the nth term of the progression, the sum of the first n terms of the progression and the infinite sum of the progression.

Triangle Solution (1) Given three elements of a triangle (SSS, SSA, or SAS) remaining angles and sides are calculated.

Triangle Solution (2) Given three elements of a triangle (ASA or SAA) remaining angle and sides are calculated. Also calculates area of a triangle.

Curve Solution Given any two parts: radius, central angle, chord length or arc length, the remaining parts, sector and sector areas may be calculated.

Polynomial Evaluation Evaluates a polynomial when values for the coefficients and x are known.

Complex Arithmetic Performs all arithmetic functions for two complex numbers.

Complex Functions (1) Calculates the following complex functions of complex numbers of the form $Z = a + ib$: Z, Z^2, $\sqrt{Z}$, 1/Z, Z^n, and $Z^{1/n}$.

Complex Functions (2) Calculates the following complex functions of complex numbers of the form $Z = a + ib$: e^z, ln Z, y^z and log Z.

Complex Functions (3) Calculates the following complex functions of complex numbers of the form $Z = a + ib$ and $W = R + id$: Z^w, $Z^{1/w}$, and $\log_z W$.

Conversions (1) Calculates length conversions.

Conversions (2) Calculates volume weight and temperature conversions.

Finance Library

Amortized Loan Schedule (Print) Prints a complete schedule of payments showing the principal and interest portions of each payment, the accumulated interest and principal at each payment, and the remaining principal balance.

Ordinary Annuity Calculates present value, (interest rate known) periodic payment or number of payments given the interest and the remaining two values.

Ordinary Annuity (Interest rate unknown) Calculates periodic interest rate given present value, payment and number of payments.

Compound Interest Computes any one of four variables (present value, future value, interest rate, number of payments) in the compound interest equation, given the other three as inputs.

Trend Line Analysis Determines linear least square fit to a set of input data points ($y = mx + b$), the correlation coefficient, and projects new points.

Sinking Fund (Interest Rate Known) Calculates future value, payment per period or number of payments given interest and the remaining two variables for an annuity compiled by equal interval payments that draws interest.

Sinking Fund (Interest Rate Unknown) Calculates interest rate required to fulfill requirements of future value, payments per period and number of payments for equal interval payment annuity.

Accrued Interest Computes the amount of interest earned, but not collected, for a given initial amount, yearly rate of interest and number of days.

Total, Average, Percent Contributions Calculates the total, average, and percent contribution of up to 16 input values.

Bond Yield Solves the discount rate which equates the present value of future interest and principal payments with the current bond market price.

Bond Present Value Determines the present value of a bond based on future interest and principal payments to yield a chosen percentage yield.

Ordinary Annuity/PV with Balloon Payment (Interest Rate Known) Knowing periodic interest rate and three of the following variables—present value, periodic payment, balloon payment payoff or number of periods, the unknown variable is computed for an annuity situation.

Ordinary Annuity/PV with Balloon Payment (Interest Rate Unknown) Given the present value, periodic payment, number of periods and the balloon payment, the interest rate for the annuity situation is computed.

Annuity Due/FV (Interest Rate Known) Knowing periodic interest rate and two of the following variables—future value, periodic payment or number of payments, the unknown variable is calculated for an annuity due situation.

Annuity Due/FV (Interest Rate Unknown) Knowing future value, periodic payment and number of payments, the periodic interest rate is calculated for an annuity due situation.

Annuity Due/PV (Interest Rate Known) Given periodic interest rate and two of the following variables—present value, periodic payment or number of payments, the unknown variable is computed for an annuity due situation.

Annuity Due/PV (Interest Rate Unknown) Knowing present value, periodic payment and number of payments, the periodic interest rate is calculated for an annuity due situation.

Annuity Due/PV with Balloon Payment (Interest Rate Known) Given periodic interest rate and three of the following variables—present value, periodic payment, balloon payment or number of payments, the unknown variable is computed for an annuity due situation.

Annuity Due/PV with Balloon Payment (Interest Rate Unknown) Knowing present value, periodic payment, balloon payment and number of payments, the periodic interest rate is calculated for an annuity due situation.

Rate Conversions Calculates nominal or effective interest rate (knowing the other) for finite or continuous compounding.

Add-On Rate Installment Loan This program calculates the finance charge, monthly payment and annual percentage rate for an installment loan.

Interest Rebate—Rule of 78's Determines the unearned interest (rebate) and the principal balance due for a prepaid consumer loan using the rule of 78's.

Straight Line Depreciation Calculates depreciation, remaining depreciable value, remaining book value and depreciation to date for the straight line method of depreciation.

Declining Balance Depreciation Computes the depreciation, remaining depreciable value, remaining book value and depreciation to date using the declining balance method of depreciation.

Sum-of-the-Years'-Digits Depreciation Calculates depreciation, remaining depreciable value, remaining book value and depreciation to date using the sum-of-the-digits depreciation method.

Crossover Point Declining Balance to Straight Line Determines the point in the depreciation schedule where the straight line method would decrease the book value faster than would the declining balance method. Remaining life of the asset and remaining book value are also calculated.

Days Between Dates Calculates number of days between any two calendar dates after the year 1582.

Variable Cash Flow (Future Value) Determines rate of return required to obtain a desired future value of a series of variable cash flows.

Variable Cash Flow (Present Value) Calculates rate of return on a capital investment based on resulting cash flows generated by investment.

Variable Cash Flow (Present and Future Value) Computes present or future value of a stream of cash flows using discrete or continuous interest computation.

Capital Budgeting Operates a generalized capital budgeting model which determines present value of a series of cash flows after depreciation and taxes.

More libraries are on the way.

TI plans to continue expanding its optional library of programs. Forthcoming subjects include: **Navigation. Surveying. Aviation.**

The PC-100 thermal printer. For the SR-56. For the SR-52. See your program at every step.

Imagine the convenience of getting a hard copy print out of: Data. Intermediate results. Answers. Imagine the efficiency of listing your entire program at the push of a key. Or, printing the calculator's *entire* data memory contents with a simple program. And now imagine seeing *every* step of your program as it's executed – both the number and the function. Imagine no more. TI's exclusive PC-100 printer is here. Ready to print long ballistic trajectories to short unit conversions. Or, complex tax analyses to simple cost/price margins.

Uses limited only by your imagination.

For technology: Plot the frequency response of an amplifier or filter circuit. Explore transient response of a complex control system. List component values in design. Or, multiple statistics: Mean. Standard deviation. Correlation coefficients. Follow the convergence of your iterative solution to a transcendental equation.

For business and finance: Print streams of cash flows and their discounted present values. Generate a complete loan amortization schedule. List sales results with trend line data. Create tables of effective yields. Investigate learning curve numbers. Compare depreciation schedules using alternate methods.

Simplifies program editing.

You can get listings fast. Just push the list key and the PC-100 prints out the SR-52's entire 224-step program memory in about 80 seconds. Less than half that time for the SR-56's 100-step memory. Identify your program code, and its position in program memory – right on the tape. Editing is easier and faster, when you can have the whole program in front of you.

See each program step executed.

Push the trace key. Now every calculation that's performed in your program is printed. The full number and the operation. Store prints out STO. Logarithms prints out LOG. Errors are a question mark (?). You can follow subroutine calls and returns. Conditional, and unconditional branches and loops. Conversions and register operations. In fact, everything your SR-52 and SR-56 can do in a program. Now you can be sure it is doing what you want it to. A real help when you're developing new programs.

Now that you can get inside a program, see it, operate on it, and edit it, you'll be sure it's running right. Giving the answers you need.

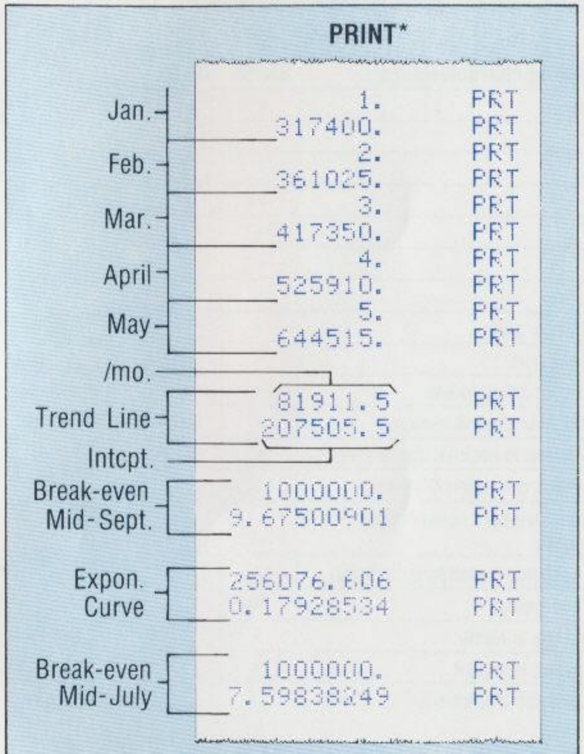
The only noise you'll hear is the quiet whirr of the print head.

TI's leadership in thermal printing technology delivers a printer that's more than quiet, and fast. It has few moving parts – mainly to drive the paper. High reliability results.

Print out results.

With the keys on the PC-100, or on the calculator keyboard. Put the PC-100 under program control and you can generate data for graphs and tables without having to look at the calculator's display.

This is the tape of the trend line problem for new product sales on page 3. The PC-100 printed the input data (months and dollars), intermediate results, and the final answers.



PRINT*
Jan.
Feb.
Mar.
April
May
/mo.
Trend Line
Intcpt.
Break-even
Mid-Sept.
Expon.
Curve
Break-even
Mid-July

```
           1.    PRT
     317400.    PRT
           2.    PRT
     361025.    PRT
           3.    PRT
     417350.    PRT
           4.    PRT
     525910.    PRT
           5.    PRT
     644515.    PRT

     81911.5    PRT
    207505.5    PRT

    1000000.    PRT
  9.67500901    PRT

  256076.606    PRT
  0.17928534    PRT

    1000000.    PRT
  7.59838249    PRT
```

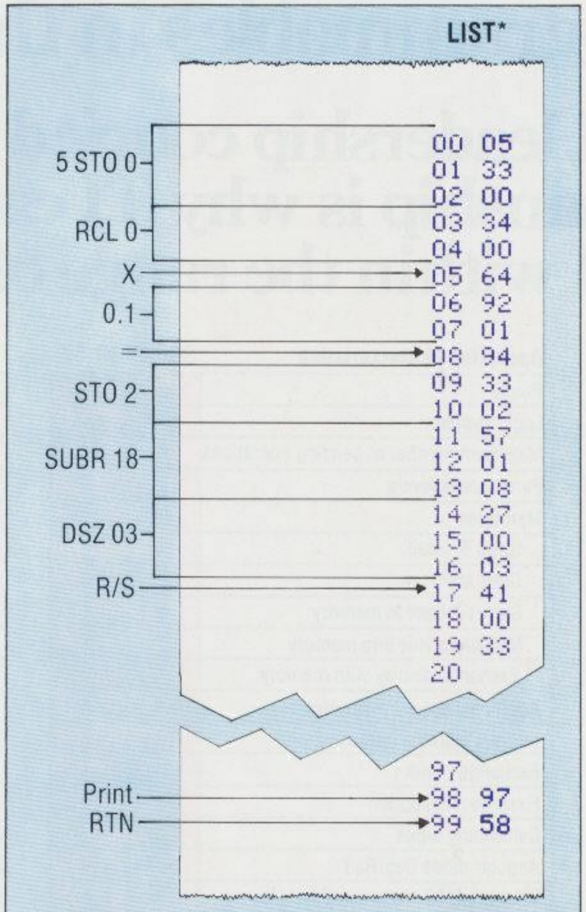
Print out of problem from page 3.

List your program.

With two keypushes on the SR-52 or SR-56, you can list your entire program. The list shows the memory locations as well as the memory's contents.

* Print-outs are shown approximately 70% of actual size.

Here's the listing of the overshoot program that was done on the SR-56 (page 11). The column on the left is the program memory location. And the right column is the two-digit key code for each instruction – its position on the keyboard (row and column). For example, 34 means third row, fourth column (RCL key).



LIST*
5 STO 0
RCL 0
X
0.1
=
STO 2
SUBR 18
DSZ 03
R/S
Print
RTN

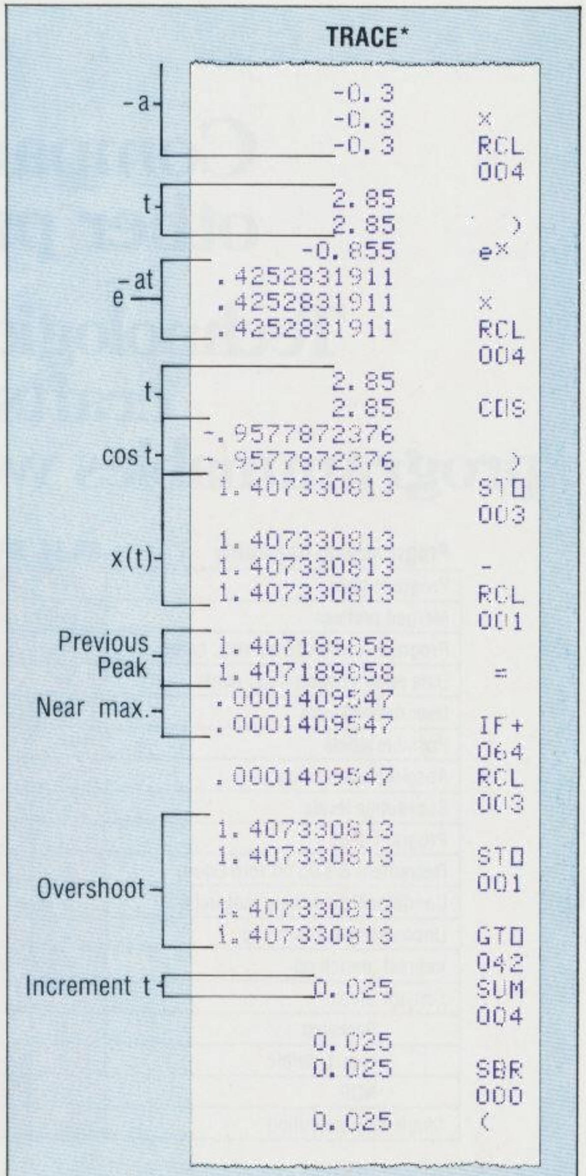
```
00 05
01 33
02 00
03 34
04 00
05 64
06 92
07 01
08 94
09 33
10 02
11 57
12 01
13 08
14 27
15 00
16 03
17 41
18 00
19 33
20

97
98 97
99 58
```

Listing of program from page 11.

Trace program execution step-by-step.

Using the TRACE key you can *see* every number that's in the calculator's display register. And the instruction as it's being performed. All right on the tape – automatically. The perfect way to debug your programs.

This tape is a trace print out of the subroutine which found the peak value in the overshoot problem (page 5). It was done on an SR-52.



TRACE*
-a
t
e^{-at}
t
cos t
x(t)
Previous Peak
Near max.
Overshoot
Increment t

```
        -0.3
        -0.3     ×
        -0.3     RCL
                 004
        2.85
        2.85      )
      -0.855     eˣ
 .4252831911
 .4252831911     ×
 .4252831911     RCL
                 004
        2.85
        2.85     COS
-.9577872376
-.9577872376      )
 1.407330813     STO
                 003
 1.407330813
 1.407330813     -
 1.407330813     RCL
                 001
 1.407189858
 1.407189858     =
 .0001409547
 .0001409547     IF+
                 064
 .0001409547     RCL
                 003
 1.407330813
 1.407330813     STO
                 001
 1.407330813
 1.407330813     GTO
                 042
       0.025     SUM
                 004
       0.025
       0.025     SBR
                 000
       0.025     (
```

Trace of problem from page 5.

Convenient to use. And key-lock security, too.

Simply remove the SR-52's or SR-56's battery pack. Press the calculator down on the connectors, turn the key, and you're ready to print. You can leave your programmable locked on the PC-100 and take the key with you.

Compare the SR-52 and SR-56 with other programmables in their class.

Technological leadership coupled with quality craftsmanship is why TI can price programmables well within the range of most professionals

Programming capability	SR-56	SR-52
Program steps	100	224
Merged prefixes	•	•
Program read/write on mag. cards	—	•
Data read/write on mag. cards	—	•*
User defined keys	—	10
Possible labels	—	72
Absolute addressing	•	•
Subroutine levels	4	2
Program flags	—	5
Decrement & skip on zero (loop)	•	•
Conditional branching instructions	6	30
Unconditional branching	3	7
Indirect branching	—	•
Editing: Step	•	•
Backstep	•	•
Insert, delete	—	•
NOP	•	—
Single step execution	•	•

Operating characteristics	SR-56	SR-52
Pause	•	—
Logic System	AOS	AOS
Maximum number of pending operations	7	10
Parentheses levels	9	9
Memories	10	22
Store & recall	•	•
Clear memory	•	•
Sum/subtract to memory	•	•
Multiply/divide into memory	•	•
Exchange display with memory	•	•
Additional special memories	1	38
Indirect memory addressing	—	•
Exchange x with t	•	—
Fixed decimal option	•	•
Calculating digits	12	12
Angular mode Deg/Rad	•*	•
Grad trig (100 grads = right angle)	•	•*
Digits displayed (mantissa & exponent)	10 + 2	10 + 2

Calculating characteristics	SR-56	SR-52
Log, lnx	•	•
10^x, e^x	•	•
X^2, $\sqrt{X}$	•	•
1/X, π	•	•
$\sqrt[x]{y}$	•	•
X!	•*	•
y^x	•	•
Int X (integer part)	•	•*
Fractioned part	•	•*
Trig functions & inverses	•	•
Hyperbolic function & inverses	•*	•*
Deg/MIN/sec to decimal deg & inverse	•*	•
Deg to Rad conversion & inverse	•*	•
Polar to rectangular conversion & inverse	•	•
Mean, variance & standard deviation	•	•*
Linear regression	•*	•*
Trend line analysis	•*	•*
Slope and intercept	•*	•*

*Programmable functions

TEXAS INSTRUMENTS
INCORPORATED