RPN-the only language that lets you "speak" with confidence and consistency to a pocket-sized computer calculator.

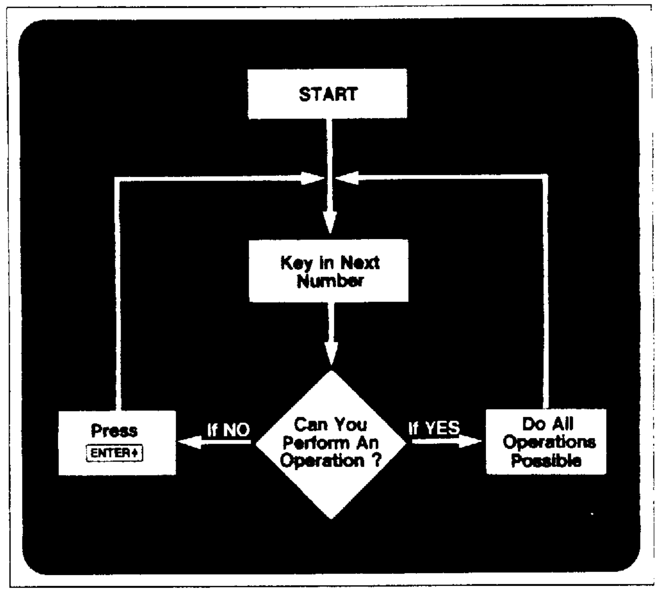
In 1967, Hewlett-Packard embarked on a major new development effort: to design a family of advanced computer calculators powerful enough to solve complex engineering/scientific problems yet simple enough to be used by anyone who works with numbers.

As part of this effort, HP carefully evaluated the strengths and weaknesses of the various languages which an operator might use to communicate with an electronic calculating device. Among those studied were:

- computer languages such as BASIC and FORTRAN,
- various forms of algebraic notation, and
- RPN (*Reverse Polish Notation*), a parenthesis-free but unambiguous language derived from that developed by the Polish mathematician, Jan Lukasiewicz.

As might be expected, each of these languages was found to excel in a particular application. For its biggest programmable desktop calculators, HP selected BASIC. For its other powerful desktop calculators, with less extensive storage capacity, HP chose algebraic notation.

But, given the design constraints of a pocket-sized scientific computer calculator, RPN proved the simplest, most efficient, most consistent way to solve complex mathematical problems.



The RPN method consists of four, easy-to-remember steps. Once learned, it can be applied to almost any mathematical expression.

Only RPN offers these powerful advantages

Compared to alternative logic systems, Hewlett-Packard believes that only RPN—in combination with a 4-register operational memory stack—gives you these powerful advantages.

1. You can always enter your data the same way, i.e., from left to right—the same way you read an equation. Yet, there is no need for a parenthesis key; nor for a complicated "operational hierarchy."
2. You can always proceed through your problem the same way. Once you've entered a number, you ask: "Can I perform an operation?" If yes, you do it. If no, you press [ENTER↑] and key in the next number.
3. You always see all intermediate answers—as they are calculated—so that you can check the progress of your calculation as you go. As important, you can review all numbers stored in the calculator at any time by pressing a few keys. There is no "hidden" data.
4. You don't have to think your problem all the way through beforehand unless the problem is so complex that it may require simultaneous storage of three or more intermediate answers.
5. You can easily recover from errors since all operations are performed sequentially, immediately after pressing the appropriate key.
6. You don't have to write down and re-enter intermediate answers, a real time-saver when working with numbers of eight or nine digits each.
7. You can communicate with your calculator confidently, consistently because you can always proceed the same way.

If all this sounds too good to be true, bear with us—you'll soon get the chance to see for yourself. But first, we need to describe how RPN and the 4-register operational stack operate.

The RPN method—it takes a few minutes to learn but can save years of frustration.

Yes, the RPN method does take some getting used to. But, once you've learned it, you can use the RPN method to solve almost any mathematical expression—confidently, consistently.

There are only four easy-to-follow steps:

1. Starting at the left side of the problem, key in the first or next number.
2. Determine if any operations can be performed. If so, do all operations possible.
3. If not, press [ENTER↑] to save the number for future use.
4. Repeat steps 1 through 3 until your calculation is completed.

A diagram of the RPN method is shown above.

Simple arithmetic, the RPN way.

Just to show how it works, let's try the RPN method on two simple problems (we'll use them again in the comparisons that begin on the next page).

Problem: $3 \times 4 = 12$

RPN solution:

Step	Press	See Displayed
1. Key in first number.	[3]	3
2. Since only one number has been keyed in, no operations are possible. Press [ENTER↑].	[ENTER↑]	3
3. Key in next number.	[4]	4
4. Since both numbers are now in calculator, multiplication can be performed.	[×]	12

Problem: $(3 \times 4) + (5 \times 6) = 42$

RPN solution:

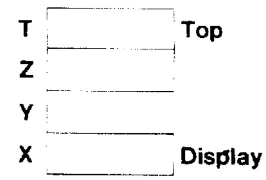
Step	Press	See Displayed
1. Key in first number.	[3]	3
2. No operations possible. Press [ENTER↑].	[ENTER↑]	3
3. Key in second number.	[4]	4
4. Since both numbers are in calculator, first multiplication is possible.	[×]	12
5. Key in next number. *(First intermediate answer will be automatically stored for future use.)*	[5]	5
6. No operations possible. Press [ENTER↑].	[ENTER↑]	5
7. Key in next number.	[6]	6
8. Second multiplication is possible since both numbers are in calculator.	[×]	30
9. Addition is possible since both intermediate answers have been calculated and are stored in 4-register operational stack.	[+]	42

If you've followed us this far, you've noticed two important facts:

1. Both of these problems were solved in the same, consistent manner, using the same simple set of rules.
2. All intermediate answers were displayed as they were calculated, and stored and retrieved as needed to complete the calculation. With RPN and a 4-register operational memory stack, there is almost never a need to write down intermediate answers.

How the operational stack works.

The four registers of HP's exclusive operational stack can be represented by the following diagram.



When a number is keyed in, it goes into the X register for display. Pressing the [ENTER↑] key duplicates the contents of the X register into the Y register and moves all other numbers in the stack up one position.

When an operation key ([+], [−], [×], [÷], [x^y]) is pressed the operation is performed on the numbers in the X and Y registers, and the answer appears in the X register for display. Numbers in the other registers automatically drop one position.

To demonstrate these points, we'll show what happens to the stack as we solve the problem: $(3 \times 4) + (5 \times 6) = 42$.

T									
Z						12	12		
Y		3	3		12	5	5	12	
X	3	3	4	12	5	5	6	30	42
	[3]	[ENTER↑]	[4]	[×]	[5]	[ENTER↑]	[6]	[×]	[+]

As you can see, all numbers are automatically positioned in the stack on a last-in-first-out basis, in the proper order for subsequent use.

Now that we've described how RPN logic operates, we can proceed with our problem-by-problem comparison of this system versus two others used in today's scientific pocket calculators.

We think you will find it interesting.