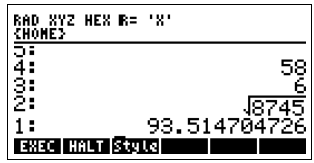
Chapter 4

The Stack

The HP 49G keeps a record of the objects you enter and the results of your operations. In algebraic mode this record is called *history*; in RPN mode it is called the *stack*.

Using the stack

Entries on the stack are numbered (as in the example at the right). An entry on the stack is referred to as being on a particular *level*. The level is the number of the line on which the entry appears. In the example at the right, 58 is on level 4, 6 is on level 3, $\sqrt{8745}$ is on level 2, and so on.



If you are working in RPN mode, you will use the stack to perform calculations. In doing so, you need to be aware of differences between the way calculations are performed and displayed in RPN mode and in algebraic mode. In RPN mode:

- A command that requires arguments—that is, a command that needs objects to act on—takes its arguments from the stack. Therefore, the arguments must be present before you execute the command: one argument per level and in the correct order. (There is one exception: when the command takes only *one* argument, you can execute the command with the argument on the command line, not on the stack.)
- A command's arguments are removed from the stack when the command is executed. The arguments are replaced by the result of the calculation.

In contrast, in algebraic mode you specify your arguments *after* you enter the command, and the command and its arguments are retained in history together with the result of the calculation.

For example, to find the cube of 52, you need to specify two arguments: the number (52) and the index (3). In algebraic mode, you enter:

52 y^x 3 ENTER

In RPN mode, you enter:

52 ENTER 3 ENTER y^x

In other words, in RPN mode 52 and 3 are entered onto the stack before the command is entered: 52 must be on level 2 and 3 on level 1 before the command is executed.

Strictly speaking, the last (or only) argument does not need to be on the stack before you execute a command in RPN mode. You can execute a command with the last (or only) argument still on the command line. Therefore, the second ENTER in the example immediately above can be omitted. However, any argument still on the command line when a command is executed will not be displayed on the stack if you undo the command (which you can do by pressing ↱ UNDO). Therefore, if you think you may need to undo a command and be able to see all the arguments, you should put *all* the arguments onto the stack before executing the command.

Example stack calculations

Using a one-argument command

1. If the argument is not already on level 1 of the stack, enter the argument onto the command line (and, optionally, onto the stack). If the argument is already on level 1 of the stack, go straight to step 2.
2. Execute the command.

Example: To calculate $\frac{1}{\sin 30}$

1. Enter 30 and press ENTER.
2. Press SIN.

   The result of sin 30 is now on level 1 of the stack. This result can be used as the argument of a further command without the result needing to be manually entered.
3. Press $1/x$.

Note that if you get a symbolic answer when you wanted a numerical answer, press ↱ →NUM. The symbolic answer is evaluated.

Using a multi-argument command

Method 1

1. Enter the arguments, pressing ENTER after each one.
2. Execute the command.

Example: To calculate 23×97

1. Enter 23 and press ENTER.
2. Enter 97 and press ENTER.
   23 is now on level 2 of the stack and 97 is on level 1.

3. Press ×.

In this example, the order in which you enter the arguments does not affect the answer. However, this is not always the case with two-argument commands. In the cube example on page 4-2, the result of entering the 3 before the 52 is the 3 raised to the power of 52, a very different result to 52 raised to the power of 3. Other examples where the order you enter the arguments is important include subtraction, division, and the percentage commands (%, %CH, and %T).

Method 2

In method 1 above, each argument is entered onto its own level of the stack before the command is executed. Another way is to enter all the elements onto the command line separating each with a space. Either:

- press ENTER to place the arguments onto the stack and then execute your command or
- execute your command with the arguments still on the command line.

Example: To calculate $\sqrt[3]{531441}$

1. Enter 531441 SPC 3
2. Press ENTER.
3. Press → $\sqrt[x]{y}$.

Step 2 can be omitted if you will not want to undo the command and see the arguments. Pressing → UNDO without having first placed the arguments on the stack deletes all record of the command: the result and the arguments. On the other hand, if you place the arguments on the stack before executing the command, pressing UNDO deletes the result but redisplays the arguments.

Multi-command calculations

Because the result of a calculation is retained on the stack, you can easily perform complex calculations by accumulating the results of sub-calculations on the stack and then treating these results as the arguments in a further calculation.

Example: To calculate $13^2 - (17 \times 19) + \frac{3}{7}$

1. Enter 13 ↰ x^2.

   The result—169—appears on level 1 of the stack.
2. Enter 17 and press ENTER.
3. Enter 19 and press ENTER.
4. Press ×.

   The product of 17 and 19—323—appears on level 1, and the previous result—169—is now level 2.
5. Press −.

   The two previous results—169—and 323—are now treated as the arguments in a further operation. This operation replaces the arguments with the result of the operation, that is, the difference between the first result and the second.