4.3. The role of examples for learning how to execute procedures

We believe that examples can give the learner the most concrete, most specific picture of exactly what to do while executing a procedure and making the necessary adjustments to specific task situations. The procedures that benefit most from exemplification are those that involve the interpretation of a general rule. An unelaborated rule, with its special notation, variables, symbols, general terms, etc., is usually too abstract for learners to comprehend. In this section, we will describe various ways in which examples can facilitate execution, including clarifying the spirit of a rule and providing a model for future solutions.

4.3.1. Learning to integrate a collection of operations: command editing

Command editing, a procedure introduced in the last section, is not a difficult concept to grasp; however, whether or not it is easy to execute depends on the nature of the system implementation. Both the DOS and VMS operating systems have command-editing features, but the implementations differ in several important respects, such as providing external cues to relevant operations. For example, on the IBM-PC, the key that recalls the previously issued DOS command is the F3 button. Of the ten function keys on the keyboard, there is no obvious reason why the desired key should be F3. Once the previous command is recalled, the user may edit it using a key labeled 'Ins' that toggles the system between insert and overwrite modes, and a key labeled 'Del' that puts the machine into delete mode. In contrast, it may be easier to remember how to initiate command editing in VMS because there are up-arrow and down-arrow keys that are uniquely associated with going back over a buffer of previously issued commands. Once a command is recalled, however, it may be harder to remember how to toggle from insert mode to overwrite mode because there are no overt function keys; the relevant sequence of keystrokes is the non-mnemonic control-a.

Command editing thus consists of a collection of operations for viewing and retrieving items in a buffer and changing the mode or state of the computer. The operations themselves are fairly simple: usually consisting of a single keystroke. As a result, the execution component of command editing is not easily described with a general rule; rather, learners must remember all the component operations and determine how to sequence them. Situational examples that show learners a complete interaction should be very valuable.[4]

The most difficult aspect of learning to execute this type of procedure may be remembering the arbitrary association of a key and a function. An

[4]The situational example about command editing presented earlier (example 12) provides much of the necessary description. To have fullest effect, the editing operations should probably be described more fully and should be set off from the body of the text.

additional benefit of examples may be to strengthen memory traces for such associations through repetition in a concrete context.

4.3.2. Learning to generate instances of a rule: renaming files

Among the types of procedures that are hardest to learn to execute are those that require the learner to generate a particular instance of an abstract rule. Examples of these procedures were provided earlier: learning to perform a *t*-test, learning to issue computer commands, or defining a function in a programming language. As discussed above, executing this type of procedure requires that the learner remember the name(s) of the procedure, the details of the rule or sequence of operations and how to assign values to any variables that appear in the rule.

Examples can help people learn this type of procedure in several ways. Consider the following typical example that was intended to help learners parse a rule, in this case, a rule for renaming files on the IBM-PC. In the manual, the example in (15) follows the general rule in (14).

REN[AME] [*d*:][*path*]*filename*[.*ext*] *filename*[.*ext*] (14)

For example, the command:

REN B:ABODE HOME (15)
renames the file ABODE on drive B to HOME.

The example clarifies some notational aspects of the rule. Elements that appear in square brackets in the rule are optional; in the example, the last three letters of the name of the command and the path are omitted. One problem with the example is that it does not clarify under what conditions the optional elements can be omitted. A series of situational examples that contain different combinations of optional elements might be necessary to illustrate these points. The example does begin to illustrate the distinction between constant terms and variables. Elements in the rule that are printed in italics are variables. In the example, the italicized elements have been replaced. The *d*: is replaced by B: and the first instance of *filename* is replaced by ABODE.

One serious problem with this example is that the filenames ABODE and HOME do not seem very typical of real filenames and, more importantly, they do not signal which is the *old* name and which is the *new* name of the file. If learners have trouble figuring out and remembering the order of the arguments in the rule, remembering this example is unlikely to help them. Some manuals attempt to solve this problem with examples like the following:

RENAME OLDFILE NEWFILE (16)