SECTION 4:
PROGRAMMING VALDOCS

Valdocs is a *language*. It is the language of *documents* of all types.

Like any language, Valdocs is composed of *words*. And every single word has the *same* meaning in Valdocs as it does in English. Not Martian. Not Computerese. English.

In English, the words "boy", "dog" and "house", represent real objects, and are called *nouns*. The specific nouns included in Valdocs are those which represent the objects you will handle with your computer: words, letters, sentences, documents, files, numbers, pictures, etc.

The verbs of Valdocs enable you to do *actions* related to creating, changing and maintaining documents or the squiggles we put on them. Typical of these verbs are PRINT, MAIL, COPY, DELETE, MOVE, RETRIEVE, FORMAT, etc. They mean *exactly* what they say.

Because the words of Valdocs are *the same as* in English, you probably know a lot of the vocabulary (Especially in familiar territory. I hardly expect a fiction writer to be very familiar with the vocabulary of mathematics....). The only real difference between a beginner and an expert in Valdocs (or any other language) is that the beginner doesn't know *where* to find the words he needs.

In human language, words are stored in your mind. You can string them together into sentences with the speed of thought. "Sentences" (tasks, jobs) in Valdocs are "spoken" by pointing at the keyboard with your finger -- or at a menu on screen with a movable pointer.

As you become fluent in Valdocs you will gain the ability to string the words of Valdocs together in "sentences" of more and more ingenuity.

WHAT IS PROGRAMMING?

If pointing is the equivalent of spoken language, programming is the equivalent of *written* language.

When *anyone* programs a computer in *any* language, all he is doing is writing down a sequence of operations which the computer is to carry out. The sequence may be short or long, simple or complex, simplistic or subtle, but it is still nothing but a series of *commands* which will be carried out by the computer one after the other.

The commands that you use in the day to day operations of Valdocs can be written down into "sentences" of commands using the program called DEFKEY.

DEFKEY

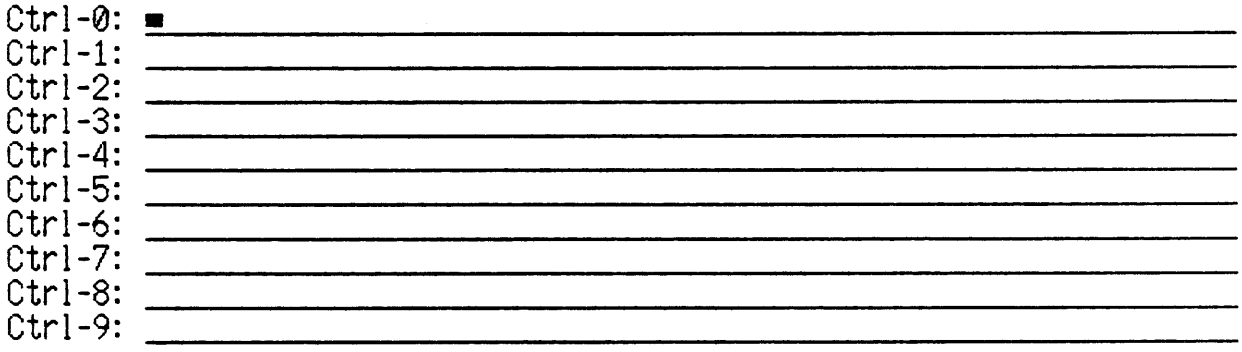
Defkey may be accessed via the MENU key, or directly from certain programs.

When accessed via the MENU key, Defkey's first menu offers you the choice of defining keystrokes for TPM, Non-Valdocs Applications, and Valdocs.

For TPM and Non-Valdocs Applications, DEFKEY is used to assign definitions to the System Controls, File Controls, etc. at the top of the keyboard. While this has many uses, it is not the subject of this Section.

When Valdocs is accessed directly from a Valdocs application, or when the VALDOCS selection is made from it's first menu as described above, Defkey presents the same set of Document and Interaction windows.

For the purposes of this Section, enter the Valdocs EDITOR, and press CONTROL + U. The following Interaction and Document windows will appear:



To achieve a degree of familiarization, please follow these steps:

1. Notice that your cursor is in the Document window.

2. Directly under the window notice that the system tells you that 254 characters remain for VALDOCS definitions. This means that a total of 254 keystrokes can be recorded in any single definition.

3. Type something on the first line. Anything at all.

4. Notice that the number of characters remaining decreases as you enter each character.

5. Move the cursor around the screen with the arrow keys.

6. You can enter characters anywhere at all, as long as you have characters remaining.

7. Press the **RETURN** key. Note that a "♪" character is entered on screen. Press **RETURN** a few more times and note that the "♪" appears as often as you press the **RETURN**.

8. If at any time you press **UNDO**, you will immediately go back to the Editor, and all characters you entered will be lost. Press **UNDO** now.

9. Press **CONTROL + U**. Note that the text you entered is gone and you are returned to line 0.

10. Type in "Mary had a little lamb". Then press **STORE**.

11. Again you are returned to the editor. But now press The **CONTROL** key plus the **[ZERO]** key on the numeric keypad.

12. Notice that the text "Mary had a little lamb" is entered at the cursor location. Press **CONTROL + [ZERO]** again and notice that the text is entered once more. Repeat this a few times. Move your cursor inside the text and repeat the command and notice that each time the text is inserted or replaced at the place where the cursor is located.

13. Enter **CONTROL + U** again.

14. Now, move to the line for Control 1, and enter: "whose fleece was white as snow."

15. Move back to the Zero line, and move your cursor to the very end of the line.

16. Find the [+/-] key. On newer keyboards, this is in the upper right corner of the numeric keypad. On older keyboards [+/-] is one of the typing keys (located on the top row next to the MAR REL key).

17. Press this key. Notice that a "\" appears in reverse video.

18. Now press the number "1". Notice that the "\" changes to a "]", followed by a "1". Then press STORE again.

19. Inside the editor, once again press CONTROL + [ZERO]. Note that both line 0 *and* line 1 are now printed out. This is called linking or chaining. Using this technique you can write commands up to the full 254 characters.

20. And now; notice your first *bug*. Unless you were clever enough to spot this one yourself, the words "lamb" and "whose" are joined together because you didn't leave spaces after "lamb" on line 0, and before "whose" on line 1. Press CONTROL + U again.

21. Make sure you're in INSERT mode (press the insert key if need be). Insert a space in front of the word "whose". Now move to the end of that line, insert *another* space, and press [+/-] followed by [ZERO]. Now press STORE.

22. Once again press CONTROL + [ZERO].

23. Notice that your sentence now repeats endlessly, over and over again. This, as you might be able to guess, is called a *loop*.

24. Press the STOP key to halt the action.

Rule: The STOP key halts the execution of a DEFKEY sentence.

25. Press CONTROL + U again.

26. Press [+/-], and then the STORE key. Notice that you are not returned to the editor, instead, characters are entered on the defkey line. Press [+\-] again, followed by a Right Arrow. Same thing. Press [+\-] MAIL, [+\-] UNDO etc. etc.

Rule: When preceded by a [+\-], most of the HASCI command keys, or editing group keys, including the numbers 0 through 9 on the numeric keypad, may be included in a Defkey definition.

This means that Defkey does not limit you to simple text; you can construct much more complex commands to suit your needs. *(Note: for technical reasons, the LEFT DELETE KEY cannot be included in definitions at this time.)*

27. Delete the sample commands you just entered, and instead, enter this sequence:

Press Keys	Effect
[+\-] BOLD	Turn BOLD on.
[+\-] RIGHT ARROW	Right Arrow
[+\-] WORD	Word key
[+\-] BOLD	Turn BOLD off
[+\-] RIGHT ARROW	Right Arrow
[+\-] WORD	WORD
[+\-] 2	Loop (repeat)

Notice that you must press the [+\-] key before *every* non-text key, as well as before the numeric pad's "2".

Press STORE.

28. Move to the beginning of the "mary had a little lamb" text, (Or press CONTROL [ZERO] to enter some more!), and press CONTROL + 2.

Note that the cursor moves rapidly through the text changing every other word in the document bold! When you get to the end of the text you entered, the cursor will flicker about once per second and the keyboard will not respond to any command except STOP.

29. After pressing STOP, press CONTROL + U again.

30. Press CONTROL + STORE. The following menu will appear.

```
Enter a name of up to 8 letters for this set of definitions
on the line below, then press STORE.

________
Press INDEX to see the list of existing files on drive B10:

<  UNDO for prior menu  >                    <  4:01 P  >
```

This menu allows you to store your definitions as a named file on disk for later retrieval.

NOTE: The Epson CP/M now actually runs programs out of the Memory that had been reserved for DEFKEY Definitions. Thus, whenever CP/M is run, your DEFKEY definitions will be destroyed. If you plan to use a definition more than once, save it to disk!

Files will be stored on the current Data Disk. If you press INDEX, you will see a list of any .KEY files that are stored on this disk.

For now, enter the name "TEST" on the line and press STORE.

31. You are returned to the Editor. If you press CONTROL + [ZERO] you will note that your definitions are still in effect. In fact, when you STORE, the changes are both stored on disk *and* made available for immediate use.

    Press CONTROL + U.

32. Using the Delete keys, delete some of your current definitions. Or just type a lot of gibberish all over the existing definitions. Then press RETRIEVE.

    A menu similar to that used when you STORE'd your definitions will appear. Press INDEX, and the name under which you made a definition file will be displayed.

    Enter the name on the line, and press RETRIEVE.

    Your file will be brought in, overwriting what was on-screen. If you now press STORE these definitions will be installed.

You can actually make a defkey definition that will load another file of defkey definitions!

This concludes your familiarization with Defkey, and illustrates the basic principles that you will use when programming Valdocs.

Virtually any sequence of operations that you can do from the keyboard can be executed from defkey, and if appropriate, this sequence can be done over and over to achieve a desired result.

Here are examples of uses I have found for Defkey in my system:

1. CURRENT FILE "BACKUP" (Store the current document as a Linear Valdocs file named "WORK.VAL", and then Retrieve it again.)

2. CENTER LINE

3. CALL CENTRAL BULLETIN BOARD, PICKUP MESSAGES, SIGN-OFF.

4. INDENT ON/OFF

5. MAILMERGE

6. DELETE CURRENT FILE

The remainder of this Section will provide a detailed examination of one of DEFKEY's more ambitious uses: *Mailmerge.*

THE ANATOMY OF MAILMERGE

The end product for a mailmerge operation is:

"Printed letters, personalized with the contents of a data file."

A Mailmerge operation requires that you have data which you wish to merge. Such data can come from the addressbook or cardfile, and with the aid of a utility called CONVERT (discussed later in this manual), can also come from Peachsoft's list manager, DBASE II files, FRIDAY, or the popular DIF format.

A mailmerge operation also requires that you have a document into which to merge the data.

You must have a means by which to perform the merge.

And last, you must have the ability to print the document into which the data has been merged.

Logically, the preparation of the data to merge is not a proper application of DEFKEY. While defkey could certainly execute the steps, the steps are executed infrequently, and are likely to be somewhat different each time.

The same holds true for the generation of the document into which the merge will occur. It must be *composed.* And that composition will occur in the editor. It may be assumed that if you were going to perform merge operations often, you would develop a "library" of files which you would use.

However, the *merge* operation itself, and the *printing* of the documents are operations that are done over and over during every "merge" session -- and as such these operations are ideally suited to handling with DEFKEY.

However, let's set the stage for our defkey operations by preparing the required merge data and form letter.

THE DATA FILE

As stated, the data file for a merge operation can come from external programs such as DBASE II or FRIDAY. ASCII files from these sources can be converted to our internal data format via the CONVERT program.

The following steps will result in an appropriate file from the Valdocs Addressbook.

1. Press MAIL.

2. Select *<A>ddress book*

3. Select *<M>ail labels/merge*

   Since the data is passed in the form of a "mailing label", you need to prepared a design appropriate to the task. The rule is that each line of the mailing label counts as one data field which you can later sort and merge.

4. Select *<A>dd a new design* and make a design like the one that follows. If you are unfamiliar with the procedure, read the menus carefully and use HELP.

```
Design name: mailmerge_label_____________________________________________
Labels per row . . . . . : 1    (1-9 labels)
Label  width in characters : 35   (1-80 columns)
Label  length in lines . . : 9    (1-9 rows)
Spaces before first label  : __   (0-99 spaces)
Spaces between labels  . . : __   (0-99 spaces)
Lines between label rows . : __   (0-99 lines)

First_name_____________________
Last_name______________________
Street_________________________
P.O._Box_Suite_________________
City___________________________
State__________________________
Zip_code_______________________
Company_name___________________
Title__________________________
```

5. Once the label design is complete, select *<P>repare label/merge data.*

6. Then select *<M>ailmerge data file.*

7. You'll be asked to choose a label design to use: select the one you just made, then select as many or as few of the cards from your address book as you like. The making of the file is not particularly fast, but after it is made you can do a lot with it.

8. Under the **MENU** key, you can find a program entitled **SORT**, which will sort the file generated (MAILMERG.INF) by the MAIL program, on any number of fields in either ascending or descending order. A file by the same name is also generated by the Cardfile program.

THE MERGE DOCUMENT

Making a merge document requires a knowledge of how the merge itself will occur.

Basically, you will do a "look-for and replace" on items named ##1, ##2, ##3, etc. These will each be *replaced* with the data from one of the data files. ##1 will be replaced by line 1. ##2 will be replaced by line 2, etc.

Because you will be merging the text directly into the editor, the output will be formatted precisely. Also, because it is in our editor, you are free to use any special fonts, typestyles, justification, centering, etc. The document will be reformatted automatically!

SAMPLE LETTER

##1 ##2
##3
##5, ##6 ##7

27 June 85

Dear ##1 ##2,

I am delighted to make your acquaintance. Sometimes it's necessary to write to someone other than yourself. This is one of those times, right ##1?

I'd like to take the opportunity to thank you and Mrs. ##2 for attending our little party here in Los Angeles. It's a long way from ##5, but hopefully it was worth it.

Please give me a call the next time you leave ##6.

Best,

The Boss

Go ahead and copy this masterpiece into your editor.

9. Once the document is entered, STORE it as "Mailmerge Form" or some such, then RETRIEVE it again for conversion into a *form. Don't forget to move the cursor to the beginning or else you will have to do it in your program.*

10. Make a note of the the number or letter <N> of your current Editor "view" displayed at the lower left of the status line.

11. Press CONTROL + Q.

12. Select *<C>onfigure Editor.*

13. Toggle *<S>ame form for all views* to NO. (You want to use this form only for the current "view").

14. Select CONTROL + Q again.

15. Select *<D>efine a form.*

16. Move the cursor to the position whose number corresponds to your current view discovered in step 10 above, and press STORE.

17. Confirm your intentions and UNDO back to the editor. Now you can conduct a little experiment. Do Control M, Select ERASE, and then confirm to THROW the document away. *Note that the document is still there.*

    Your original document has been converted into a form, which reappears, after being edited, STORE'd or discarded, in it's original condition.

MERGING DATA

1. Do a CONTROL + W to position the cursor at the beginning of the document.

2. Now do a CONTROL + L.

3. Select *<D>ata Merge.*

   Note that the column on the left, the *Look-fors*, are the familiar ##1, ##2, etc. items you inserted into your form. In the right column, the Replace items are the items from the first record in your data file, if the file (MAILMERG.INF) is on the same data drive.

4. Select *<M>erge with current document.*

5. Press UNDO twice.

6. Do a CONTROL + W to the beginning of the document. Note that the information from the first RECORD has been merged into your form. If the form was laid out correctly, you should have a very attractive display.

PRINTING THE DOCUMENT

This step is simplicity itself.

1. Press PRINT.

2. Select *<D>isplayed Document.*

3. The system will proceed to print the document you just created.

4. Do a CONTROL + M, E RETURN, T RETURN, to throw away this document, because you are finished with this copy of it. (Of course, you *could* store it if you wished.)

5. Your *Form* will reappear all fresh in the document window.

PUTTING IT ALL TOGETHER WITH DEFKEY

Now, you *could* just sit there and do all this manually if you wished. And if you only had a few letters, why not? But programming Valdocs with Defkey is the point of all this, so lets set up a DEFKEY sequence of keystrokes that will take us from a form to a printed document and back again.

1. Press CONTROL + U.

2. Select an empty or available slot.

3. Enter the following keystrokes:

```
CONTROL + L     (Enter Look-for Menu)
D RETURN
N RETURN        (Next record)
M RETURN        (Merge with document)
[+\-] UNDO
[+\-] UNDO      (Back to editor)
[+\-] PRINT
D RETURN        (Print the displayed document)
CONTROL M
E RETURN
T RETURN        (Throw it away)
[+\-] X         (Do this again)
```

X is the slot number from the numeric keypad into which you entered this definition.

4. Press CONTROL + STORE. Save this definition for a future use.

5. Back in the editor, first throw away the document you merged before and get a clean copy of the form on screen. Then press CONTROL + the number of the slot you put your definition in -- and watch.

If you entered everything correctly, you should see the computer proceed to do the job you just programmed it to do -- it should start merging and printing letter after letter. If everything seems to be working -- sit back and enjoy.

(Of course, it *is* possible that you entered something wrong -- or that some subtle point of all this escaped you. If things arn't working the way you want, you have what we call a *bug* -- good hunting!)

That concludes your introduction to programming Valdocs. In only *24 keystrokes,* you wrote a program that would normally take a programmer weeks or months to accomplish.

During the coming months we'll be adding lots of new features to Valdocs. But now, every time we add a feature -- we add to the "vocabulary" of your *language,* enriching the possible applications. At the same time, Defkey will receive enormous improvements of its own. *Much* larger definitions will be possible. Elegant control structures will be added. And we're working on ways to suppress display of menus so as to speed up execution of your programs.

Oh, and have fun.....