STANFORD RESEARCH INSTITUTE
Menlo Park, California 94025 · U.S.A.

NLS USER TRAINING GUIDE

A Self Teaching Introduction to the oNLine System

20 APR 77

Augmentation Research Center
STANFORD RESEARCH INSTITUTE
MENLO PARK, CALIFORNIA 94025

This document was written by Beverly Boli, Ann Weinberg, and Nina Zolotow. It was produced in partial fulfillment of the Rome Air Development Center (RADC) National Software Works contract number F30602-75-C-0320.

TABLE OF CONTENTS

INTRODUCTION

The NLS USER TRAINING GUIDE is designed to help you use the oNLine System (NLS) to compose, transcribe, and edit text; set up files; and send, receive, and print documents. This GUIDE introduces you to the system and shows you how to put some of its features to use.

The GUIDE is a collection of instructional documents. The first is the "Preface to NLS," which introduces you to the basic concepts and jargon necessary to use the system. The remainder of the documents are called "Sample Sessions." Each session includes a brief introduction, an instructional section that takes you step by step through the specific commands necessary to perform certain tasks, and a summary of all of the commands and concepts taught in that session. An exact presentation of what you type in to execute the commands and what you will see at your terminal, along with general explanations, should make the sessions self-instructonal.

PREFACE TO NLS

NLS User Training Guide

INTRODUCTION

The oNLine System (NLS) is an interactive computer system that provides a variety of ways to enter text, highly flexible editing, formatting and printing, publication management aids, communication through the computer, catalogue facilities, and connection to other systems. NLS is an integrated collection of tools for anyone who does "knowledge work"--anyone who needs to research, organize, communicate, reshape, condense, transmit, or present information.

One way of understanding what you can do with NLS is to look at a typical knowledge worker using the system. Let us take the case of a researcher using NLS to both gather his information and present it in final printed reports. Every tool the researcher needs to produce camera ready copy of his reports is available through NLS. He may research information stored and catalogued in the computer, use the system to keep notes or organize his material, and communicate with his fellow-workers through the computer (e.g., sending memos that are permanently recorded and catalogued, sharing files, even "talking" to others by means of the keyboard).

All drafts of his reports are easily written and edited online, including moving or copying any portion of text from one file to another (automatingthe traditional cut and paste technique) and incorporating linework graphic illustrations into the docment. Several tools enable the writer to control his document; for example, the date, time, and name of the person who made the last change in any paragraph is recorded. When the final draft is ready (or any other time in the life of the document), the writer may experiment with formats for a line printer or phototypesetter. He may proof photocomposed pages with approximate fonts and type sizes on graphic display terminals. When he is satisfied, he can automatically send the document to a phototypesetter.

As you work with NLS, you will need to learn some terminology and ways to do things that may be quite new and different. This Preface gives you simple definitions of some basic terms (which are shown in quotes the first time they appear), explains how to tell the system what you want to do and how the information you put into the computer is organized for you, and lists some special characters you will want to know about. It also introduces new users to Typewriter NLS (TNLS) and Display NLS (DNLS). These sections are followed by a list of NLS "subsystems"--sets of related NLS commands. If you want to learn more about NLS on your own, the last section tells you how to use the Help services provided by NLS.

HOW FILES ARE ORGANIZED

We say that you are "online" when you are working at a terminal that is hooked up to a computer. To work easily and productively, it is important to understand something about how information is organized in NLS. Breaking information into different kinds of units makes it easier to manipulate online. The basic unit for organizing information is called a "file", which you create and name. A file is a work space reserved in the computer where you store information according to some classification useful to you. Your file may be an article, letter, program, data base--anything you want. Any information that you type in at your terminal will go into a file.

Files are kept in "directories". A directory is like a private library of files (or your own file cabinet). Most users have a personal directory containing all of their files and may share other directories with co-workers.

Within a file we further organize information into two classes: "strings" and "structures" (Figure 1).

A string is one or more consecutive characters. Three kinds of strings are commomly used in NLS.

"Characters" are single elements that can be visible or invisible, such as a letter, number, punctuation mark, space, or carriage return. Characters with special functions, called "control characters", are described later.

"Words" are continuous groups of letters and/or numbers bound by spaces and/or punctuation marks. The system recognizes that punctuation marks are not part of a word.

"Text" is a group of one or more contiguous characters, visible or invisible, that you define by indicating the first and last character.

The term structure is used in two ways in NLS. By "file structure" we mean that NLS allows you to structure your information in an hierarchical outline, with major ideas or topics followed by supporting points or subtopics beneath them. With this kind of organization you can handle logical sections of a file with single commands. For example, you may want to move a heading and all of the information under it from one part of a file to another. You can do this by specifying the right kind of structure. Four of these structure types are defined in NLS.

A "statement" is made up of strings. It may be a single character, a word, a title, some text, or a paragraph. Statements are the basic structural units of a file. They are the building blocks that make up the other three structure types. This paragraph is a single NLS statement.

A "branch" is any statement plus all its substatements, all their substatements, and so on to the end.

A "group" is a series of consecutive statements (including all their substatements) at the same "level" (described below). You define a group by indicating the first and last statements in the group.

A "plex" is all the branches on the same level with the same source. The difference between a plex and a group is that a plex is all of the consecutive branches at one level, while a group can be any contiguous set of those branches. In Figure 1, notice that statement 1 has three branches below it. We say that statement 1 is the "source" for those branches beneath it. All of the statements directly below 1--that is, 1A, 1B, and 1C--make up a single plex.

"LEVEL" refers to the relative position of a statement in this outline structure. For example, if you have a heading followed by three paragraphs beneath it, you would think of the heading as being at a higher level than the three paragraphs below it. This idea might be clearer if you refer to Figure 1. Moving from statement 1 to 1A is called going "down" a level; we say that statement 1A is a "substatement" of 1. Moving from statement 1B3 to 1B is going "up" a level, because 1B is higher in the outline than 1B3. As you add to a file, you will be able to indicate at what level you wish to write.

Statements are automatically numbered (as shown) when they are entered in a file. Each file begins with a special statement called the "origin" statement (the line numbered 0 in Figure 1). The system creates the origin statement, which includes the name of the directory, the name of the file, and some other housekeeping information. The origin statement, Statement 0, is at the highest level of the file outline and is the source for all other statements in the file.

NLS COMMANDS

You manipulate the computer system by issuing "commands." Commands are the way you specify what you want the computer to do, such as delete a word. The general form or arrangement of all the steps needed to complete a command is called "command syntax".

You complete about the same steps for most commands. The first word of a command is usually a verb, followed by a noun. After this you will probably specify an ADDRESS, such as the location of a word you want to delete. After this, commands vary. For example, if you are deleting a word you simply confirm the command. If you are inserting a word, you might type in some text at this point. Here is how these commands look:

```
Delete Word ADDRESS OK
Insert Word ADDRESS TYPEIN OK
```

Figure 2 illustrates a typical NLS command, explaining what is expected from the user at each step. Note that specifying commands differs slightly between Typewriter NLS (TNLS) and Display NLS (DNLS).

To help you know what is expected from you while you are specifying a command, NLS prints a symbol called a "prompt". For example, "C:" is the general prompt for a command word, such as "Delete." If more than one choice is possible, you are prompted for both of them (e.g., B/A:, BUG or ADDRESS). Figure 2 illustrates only the basic prompts, known as "partial prompting". Some commands include choices you would rarely make; these show up with "full prompting".

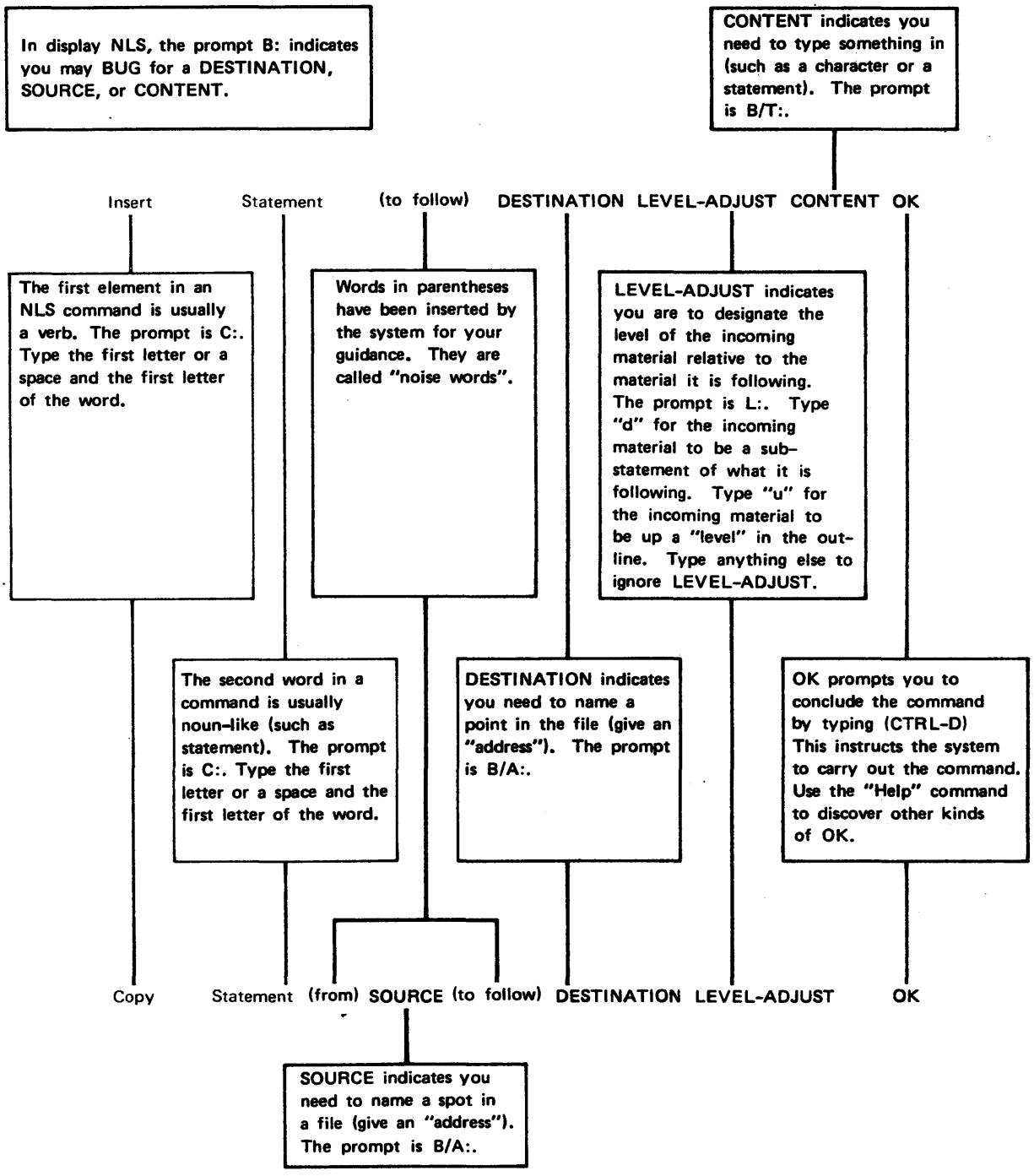

FIGURE 2 COMMAND SYNTAX

LOOKING AT FILES ONLINE

Changing Your Location

When you enter NLS, you will always be located at the same place in a file called your "initial file". In most cases you will change your location in two different ways. In TNLS, every time you specify a new command with an ADDRESS in it, you are moved to a new location. (This happens less frequently in DNLS because your location is always clearly displayed on the screen.) You can also use several NLS commands designed just for the purpose of moving you from place to place. Some of these commands enable you to move to locations specified in "links".

A link is like a bibliographic citation, telling you where to find something online. It appears as text in a statement--"linking" you to some other text in another location. The completeness of the information within a link may vary, but a full link will look like this:

<DIRECTORY, FILENAME, IN-FILE-ADDRESS : VIEWSPECS>

IN-FILE-ADDRESS means any address within a file, such as a statement number. Notice that a comma follows the name of the directory and file, and an angle bracket is at either end of the link. "Viewspecs", described below, are preceded by a colon. If you are not including viewspecs, you may leave out this colon.

Below are three examples of links, illustrating the variations in the information they may contain. The first includes something in each field (an ADDRESS consisting of text in quotes points to the location of that text in the file); the second leaves out the directory name (a common practice when the link and the file to which it points are in the same directory) and viewspec fields; and the third is very short, linking to a statement number in the same file in which the link is located.

<Calendar, Events, "Bastille Day" : t>

<Birthdays, 3>

<1a>

Links are used in many ways. The Sendmail subsystem informs you about long items other users have "mailed" to you by sending you Journal links to them. Typical Journal links look like this:

<DJOURNAL, 20001, 1:w> and <JOURNAL, JRNL42, J37000:gw>. You may tell co-workers how to find things you are working on together by giving them links to locations or leaving links in your files that they may use. There is an NLS command that will automatically take you to the location specified in a link when you point to that link.

Changing Your View

In NLS you can do more than move around in your files to see different parts of them. The hierarchical structure of NLS files permits you to "view" your files in several different ways. You can control many aspects of how your file will appear to you on an output device (such as your typewriter terminal, the display screen, or a line printer) with single character codes called "viewspecs".

By specifying the viewspec "t", for example, you can print out or display only the top line of each statement in your file. This will give you a quick outline view of the entire contents of your file. You might, instead, specify that only the first two or three levels in the file structure be shown. This would have the effect of presenting the major headings or ideas in the file. Other viewspecs allow you to do some formatting in the file, such as showing the file with blank lines between each statement, or turning off level indenting.

USING TNLS

To begin working in TNLS you "log in", or sign on to the system. This simply means that you tell the computer who you are, what kind of terminal you are using, and that you would like to use NLS. Once you have completed this procedure, the system will be ready for your commands.

TNLS terminal keyboards are very much like those on normal typewriters. They may have a few keys that will be unfamiliar to you, but you will quickly become comfortable using them. The important keys to locate immediately are the CONTROL key (often represented as "CTRL") and carriage return (usually labeled "RETURN" or "CR"). The CONTROL key is used to input control characters, as described above, while the carriage return is used to confirm a command or part of a command.

When using a typewriter terminal, you need to keep track of where you are in a file. This sometimes takes a little practice if you are only used to conventional methods of working with text (paper and pencil, or typewriter). You move from place to place according to the address you specify in each command or by commands designed to change your location. An address is always an exact point in a file--that is, you are always located at a specific character.

All the examples below refer to the outline in Figure 1.

Common Forms of Address

Statement Numbers

As described above, all NLS files are organized in an outline structure. When a statement is entered into a file you allocate it a place in this structure, either above, on the same level as, or below the preceding statement. A "statement number" is assigned to each statement, representing the exact position of that statement within the structure of the file. As you saw in Figure 1, statement numbers consist of alternating numerals and letters, a scheme you are probably familiar with. A statement number does not remain a permanent part of the statement and will change if the user changes the position of that statement.

When you type a statement number as an address, you move to the first character of that statement. Thus, if you give instructions to delete statement 3a, the statement "ENVIRONMENT DEFINED:" will disappear. If you say delete word

at 3b, the word "RELEVANT" will be removed from the statement. If you say delete character at 3b, then the first character "R" will be deleted.

SIDs

Statement numbers are very effective for showing the location of a statement in a file and its position in relationship to other statements. However, they are not always suitable for editing files, because the number of a statement will change whenever its position in the file changes. For this purpose, NLS provides another kind of numbering. Statement IDentifiers (SIDs) are unique numbers automatically assigned to statements in a file in the order in which they are created. An SID remains with its corresponding statement for the life of the statement (despite editing changes). It is preceded by a 0--thus, the first statement you create in a file after the origin statement is 01, the second 02, etc. SIDs are especially helpful in editing files. Regardless of where a statement is moved in the structure of a file, it keeps the same number.

Content Addresses

When you address a statement (by its statement number or SID), you are automatically taken to the first character in that statement. By combining different address elements you can reach any character or string you choose. The easiest way is to type in the statement number, and then the text (in quotes) that contains the character you want. This takes you to the last character of the string you put in quotation marks. For example, to change "DEFINITIONS" in statement 3b1 to "DEFINITION", you could specify Delete Character at A: 3b1 "ons". The last character in this address string ("s") would then be erased.

Other Addresses Within Statements

There are many other forms of addressing in TNLS. You can search for a word, move by counting words or characters, or search for strings according to where they are in a statement. For example, a special symbol (+e) exists for the character at the end of a statement. (Refer to the NLS CUE CARD or type "infileaddress" in Help to see a list of these address elements.) You can also restrict the search to a given statement or some other structure.

USING DNLS

To begin working in DNLS you "log in", or sign on to the system. This will be a slightly different procedure from that used to log in to TNLS, but you are doing the same thing--telling the computer who you are, what kind of terminal you are using, and that you would like to use NLS. Once you have completed this procedure, the system will be ready for your commands.

You will immediately notice three major differences between DNLS and TNLS. The equipment is different and there is more of it. You can always see where you are in a file in DNLS and the results of any changes you make to that portion of the file are displayed on your screen. The way in which you address locations in files also differs from TNLS.

DNLS Work Station

The DNLS work station consists of the display and keyboard, a pointing device called a mouse, and a five-finger keyboard called a keyset. The input devices, keyboard, mouse, and keyset, are connected to the computer through a device called the Line Processor, described in detail in the "Line Processor User's Guide". Much of the speed and flexibility of the NLS command system depends upon the use of these devices.

Display and Keyboard

Although DNLS can support many kinds of displays, you will probably be using a Data Media, which has been found to be reliable and easy to use. The keyboard is like a standard typewriter keyboard, with special black keys on either side.

The display screen is divided into two areas, called the Feedback area and File Window. The Feedback area uses about six lines at the top of the screen to provide you with information about your viewspecs, "TENEX" (the time-sharing system used by NLS), your current file, and the subsystem you are using. It also displays the current command you are specifying and input for that command. The File Window, which occupies the remainder of the screen, displays files or parts of files.

Another feature of your screen is a cursor or "bug mark", a travelling mark on the screen, often a bright underline or block, used to "point to" or identify characters.

Mouse and Keyset

The mouse is a hand-sized device with three buttons on the top of it. The mouse rolls freely on any flat surface, moving the cursor on the display screen correspondingly. The buttons are used alone or with the keyset or keyboard to mean different things. For example, pressing down the right button alone confirms a command; pressing down the left and center buttons together while typing characters from the keyboard or keyset changes your viewspecs. You will quickly become comfortable with the mouse, since the movement of the cursor on the screen mirrors your hand movement--when you move the mouse to the right, the cursor mark moves to the right.

The keyset is a device with five piano-like keys for entering characters into NLS using a logical binary code (shown on the "Mouse and Keyset Cue Card"). An alternative to the keyboard, the keyset is designed to facilitate rapid editing. With your left hand on the keyset and your right on the mouse, you can give input to the system without ever moving your hands back to the keyboard. This allows you to keep your eyes on the screen while quickly specifying commands, moving around in files, and changing views. Using the keyset is optional--you may want to wait to use it until you are comfortable with DNLS.

Addressing

To address a location on the display screen you will usually point with the mouse rather than type in an address, as required in TNLS. As described above, the mouse controls the cursor or bugmark. When you want to address a location on the screen, you "BUG" that location by moving the cursor to that character and pushing down the right-most button on top of the mouse. This will mark that character with an underline, blot-out, or character highlighting. This indicates that the computer knows the position you selected. To address locations not on your screen you can use the same techniques described under TNLS addressing.

NLS SUBSYSTEMS

NLS commands are divided into subsystems, which are sets of commands related to particular activities. For example, commands for distributing and cataloguing messages and documents online comprise the Sendmail subsystem. Below is a description of the subsystems currently available to users. Your group may have some subsystems of its own you will also want to learn about.

Base Subsystem

Base is the "home base" subsystem in which you are automatically placed when you enter NLS. It has commands that allow you to read, write, and modify information online and print it in different ways, among other things.

Calculator Subsystem

The Calculator subsystem provides a variety of commands that allow you to do simple arithmetic--add, subtract, multiply, and divide--and integrate your totals into an NLS file.

Format Subsystem

The Format subsystem helps you automatically print out information in predesigned formats.

Graphics Subsystem

The Graphics subsystem enables you to write, display, and output diagrams containing line drawings and text labels. Diagrams and text of NLS statements are stored in the same NLS file and may be printed through a phototypesetting device. This subsystem requires special graphics equipment.

Message Subsystem

The Message subsystem enables you to handle communications from the TENEX message-sending facility ("SNDMSG") through NLS. With this subsystem you can move your messages into NLS, sort messages, reformat Sendmail items to correspond with those sent through SNDMSG, and automatically send messages through SNDMSG via NLS.

Modify Subsystem

The Modify subsystem contains five groups of commands that are extensions of the Base subsystem's editing commands.

Programs Subsystem

The Programs subsystem contains commands to handle special programming needs and allows you to add to the existing subsystems in NLS. In Base, you can write filters through which you may view a file, or programs which actually modify statements containing particular forms of text you specify.

Proof Subsystem

The Proof subsystem presents pages as they would appear phototypeset, to allow preliminary proofing of formats. The Proof subsystem will display the layout of the page correctly but not the type font, and it will only work on suitable high-resolution display terminals.

Publish Subsystem

The Publish subsystem aids you in document production. You can automatically generate a table of contents, references in standard formats, or an index keyed to statement numbers. You can also count words.

Sendmail Subsystem

The Sendmail subsystem allows you to send messages and documents to a list of people known to NLS and have these messages catalogued and stored in the NLS Journal. The recipients may receive hardcopy or notice of the item in their initial file, with a link to allow immediate online access, or the item itself if it is short.

Useroptions Subsystem

You can alter how you interact with NLS to fit your own equipment, use patterns, and style by specifying the parameters controlled by Useroptions. Changes made with a Useroptions command will be in effect in future NLS sessions, until you use the command again to make more changes.

HELP SERVICES

If you want more information or the answer to a question, there are several help services that you should know about. The two described below are the "?" and the "Help" command.

You may find that sometimes you are not sure what you can do next. If you type a "?" at any point, you will get a list of all your immediate alternatives. For example, if you type "i" for insert and then a "?", you will see a list of all the things you can insert. You can then specify one of the alternatives you see and it will become part of your command.

The "Help" command provides information about how to use NLS, its subsystems, programs, and commands. It gives you definitions of all the jargon terms, explains each term in context, and refers you to related terms. Help also describes how to do certain tasks in NLS, how to use commands, and points out the unexpected consequences of some commands.

To use Help, type the Help key (or <CTRL-Q>) at any point while specifying a command and you will get information about what you were doing before you typed <CTRL-Q>. You may also type the Help key at the herald, followed by a term you want information about. You can continue to type in more terms for more information. Typing a Command Delete (<CTRL-X>) will take you out of the command. For a complete description on how to use Help, see the "Help Sample Session".

DNLS INTRODUCTORY SAMPLE SESSION

NLS User Training Guide

INTRODUCTION

NLS, or the Online System, is the name of the computer system you will be using. Online means you receive immediate feedback about what you have just typed at your terminal.

NLS has facilities to let you do almost everything you need to do with text: compose it, edit it, send it to and receive it from other persons, file it in one or more categories, cite and easily obtain documents, search for documents by author and subject, search in documents by word or phrase, and print in practically any format.

This sample session introduces you to the display work station, and demonstrates the commands used for writing a memo and editing it. You will be working with DNLS, the display version of NLS. Sit at a display work station while you read through the introductory explanations and later type in the commands and text as they are described.

Although this sample session describes specific editing commands, we add notes at each step which generalize the operation. With this session as a model, the inexperienced user should be able to perform any of the operations described here and refer to other NLS documentation for more information about the system.

Throughout this sample session we spell out the sequence of keys you strike to make something happen and then show what will appear at your terminal in response. Keys that do not print, such as carriage return, command accept, and escape, are named inside angle brackets, e.g., <CR>, <OK>, and <ESC>. <SP> represents a space. The control key <CTRL> is used like the shift key. You hold it down while you type the letter that is after the hyphen. The notation for control keys is <CTRL-(some control character)>, for example <CTRL-W>.

When you see <CA> or <OK>, type the OK button or push the right most button of the mouse.

If you get stuck or confused, typing "?" will show you the next possible alternatives. You then type in one of the alternatives and continue your command.

DNLS WORK STATION

The DNLS work station consists of the display and keyboard, a pointing device called a mouse, and a five-finger keyboard called a keyset. The input devices are connected to the terminal computer through a device called the line processor, described in detail in the "Line Processor User's Guide." Much of the speed and flexibility of the NLS command system depends on the use of these devices.

Display and Keyboard

Although DNLS can support many kinds of displays, you will probably be using a Data Media, which has been found to be reliable and easy to use. The numbers and letters on the keyboard are arranged like a standard typewriter keyboard, with special black keys on either side. The functions of the special keys are listed below. A description of the way information is displayed on the screen will be given later in this sample session.

Another feature of your screen is a cursor or "bug mark," a travelling mark on the screen, often a bright underline or block, used to "point" to or identify characters.

Mouse and Keyset

The mouse is a hand-sized device with three buttons on the top. It rolls freely on any flat surface, moving the cursor on the display screen correspondingly. The three buttons can be used alone or with the keyset or keyboard to mean different things. This will be explained later. You will quickly become comfortable with the mouse, since the movement of the cursor on the screen mirrors your hand movement, e.g., when you move the mouse to the right, the cursor mark moves to the right.

To address a location on the display screen, you move the cursor under that character and push down the right-most button on top of the mouse. This will mark that character with a circle, square, underline, blot-out, or character inversion. This indicates that the host computer knows the position you selected.

The keyset is a device with five piano-like keys for entering characters into NLS using a logical binary code (shown on the "Mouse and Keyset Cue Card"). The keyset is an alternative to the keyboard and is designed to facilitate rapid editing. With your left hand on the keyset and your right on the mouse, you

can give input to the system without ever moving your hands back to the keyboard. This allows you to keep your eyes on the screen while quickly specifying commands, moving around in files, and changing views. Using the keyset is optional and takes a while to learn. You may want to wait until you are comfortable with DNLS.

Special Functions Keys on Keyboard and Mouse

KEYBOARD	MOUSE
OK or CA (Command Accept)	right button alone

Used to terminate what you type in or a viewspec, to BUG something on the screen, and to give a confirmation any time you are prompted for an OK.

BACKSPACE (Backspace Character)	left button alone

One character of input is deleted each time this is pressed. <CTRL-A> or BC.

BACKSPACE WORD	left and center buttons

One word of input is deleted each time these are pressed. <CTRL-W> or BW.

CMD DEL (Command Delete)	middle button alone

Used to abort a command. <CTRL-X>.

RETURN (Carriage Return)	no corresponding function on mouse

Used to enter a carriage return into text or to terminate TENEX commands.

ESC (Escape)	right and left buttons

Used to complete command words and file names in TENEX, filenames in NLS, and to repeat a search.

Line Processor

The line processor is a microcomputer which processes data to and from the computer. The four silver toggles on the front of the line processor should all be down. If any of the four

numbered lights at the top start to flash, press the System Reset button.

INSTRUCTION

Most users reach NLS through the ARPA Network. For the current connection procedures at your site, see someone knowledgeable in NLS. When you have made your connection you will see:

"TENEX 1.##.## SITENAME 1,##.##" and then an "@".

This is the TENEX ready signal, also called a herald. TENEX is a system within the computer that provides access to NLS and other programs. The herald tells you that TENEX is waiting for you to type.

1. To identify yourself to the TENEX system:

   ....
   You type:
   log<SP>DIRECTORYNAME<SP>PASSWORD<SP><CR>
   You see:
   @log DIRECTORYNAME
   DIRECTORYNAME
   JOB # ON TTY # DATE TIME
   ....

   If you do not know a DIRECTORYNAME or PASSWORD, ask the person in your organization who usually helps people with NLS or call (415) 326-6200, extension 3630, between 8 a.m. and 5 p.m. Pacific time.

   After the job information has been typed by the system, your login is completed and the TENEX herald "@" will again appear at the left margin. It is again your turn to type.

2. To enter the DNLS system:

   To enter the DNLS system, you type "dnls<CR>". If you are using a group directory, you will be asked to type an ident after you type "dnls<CR>". Type in the ident (yours, one that you have been given, or one that is associated with this particular directory), following it with a <CR>. After completing this, the screen will become blank and then fill with some writing. This is often referred to as "recreating". This writing, called your initial file, is your "current file." Unless you specify otherwise, all commands affect your current file, the file that is displayed on your screen. We will discuss the initial file at a later time.

```
.....................................................
You type:
          dnls<CR>
You see:
          @dnls
.....................................................
```

The screen is divided into areas, each containing specific useful information. These are described below:

The Viewspec Window is the uppermost right corner.

In this window, you will see a series of uppercase characters followed by lowercase characters. These characters indicate the current status of certain viewspecs. This section of the screen will flash or be underlined when the viewspecs can be changed. You will learn more about viewspecs in later sample sessions.

The Typewriter Simulation (TTY) Window is the two lines in the upper left corner.

This area provides feedback and shows interaction with TENEX. It is used for error messages, system messages, and the name of the file being loaded or updated. If TENEX is called with a <CTRL-C>, interaction is shown here. The TTY Window will remain empty if there is no information. When you first enter DNLS, this area will probably be blank.

The Subsystem Window is one line below the TTY Window.

This line displays the name of the subsystem being used.

The Command Feedback Window is one line below the Subsystem Window.

In this window the current command phrase is displayed with noise words and prompts. Some prompts to remember are:

C: Command Word

T: Type in

L: Level

V: Viewspec

OK: Command Accept

Y/N: Type "y" for yes or "n" for no

B: Bug character with the mouse and press the right most button

>... Executing command

The Typein Feedback Window is below the Command Feedback Window.

This area displays the typein or address portion of a command as it will be accepted after the OK.

The File Window is the rest of the screen.

The File Window displays files or parts of files. This window differs from the others in that you can use the mouse for pointing.

For each subsequent step, we will show you what to type and what you will see. Sometimes you may see more on your screen than we show you. We will be concerned mainly with the Subsystem Window, the Typein Feedback Window, the Command Feedback Window, and parts of the File Window.

3. In this sample session, you will write a memo; to do so, you will need to create an empty file (or work space) in which to put it. You give every file a name, in this case name it "memo," so that you can call it back in future NLS sessions. File names may include up to 29 letters and/or digits, and must begin with a letter. File names may not include spaces, commas, periods, or semicolons.

.....

You type:

<SP>crfmemo<OK>

You see:

```
BASE
  > C:  Create > C: File T:
memo

< DIRECTORYNAME, MEMO.NLS;1, >
```

.....

The screen should be blank except for the origin statement, statement 0, which consists of the DIRECTORYNAME and the file name. The system will add other information to the origin

statement when you update the file. When you create a new file, this automatically becomes your current file.

When NLS is ready for you to give a command, it asks you for a command word by prompting you with a C:, and when it is ready for you to type in some text, it prompts you with T:. The system usually completes the command word for you after you have typed in the first letter. In the case of some commands that are used less often, you have to type a space and then one, two, or three letters.

Note: If you are in the middle of a command and type CMD DEL (Command Delete) or the middle button on the mouse, the command will be aborted and you may begin again.

If you leave the system without finishing your work on this file (or if you create another file), you can retrieve it (or any other stored file) in DNLS by using the command Jump Link and typing in the file name followed by a comma after the T:. YOU DO NOT NEED TO DO THIS NOW, because the file named MEMO is your current file since you just created it. The command is shown here for your future use:

.....

You type:

jlmemo,<OK>

You see:

BASE
  > C: Jump > C: Link T:
memo,
< DIRECTORYNAME, MEMO.NLS;1, > (this shows first in the tty window)

.....

4. You will begin writing your memo by inserting a statement to follow statement 0 in your file MEMO. After you type the two command words "Insert" and "Statement," you will see the words "(to follow)." These words are called noise words and are typed by the system to help you understand the purpose of a command or what input is expected next from you. At this point you will move the cursor so that it points to any position under the origin statement. Press the right-most mouse button and the character above the cursor will be marked. You have now BUGGED the origin statement. This indicates to the system that the new statement you are inserting will follow the origin statement. (We use the term BUGMARK below to represent a character you have bugged somewhere in the File Window. We use the term BUG to indicate when you need to BUG something.)

```
.....................................................
You type:
          isBUG<OK>
          Contradictions have been alledged in our
          description of the elephant.<OK>
You see:
          BASE
            > C: Insert > C: Statement (to follow)
            B: BUGMARK > L:
            Contradictions have been alledged in our
            description of the elephant.
.....................................................
```

After you BUGGED statement 0 to indicate which statement you wanted your new statement to follow, you were prompted with L:. If you were creating a file that used an outline structure, you could indicate at which level you wanted your new statement to be. In this sample session, you can ignore L: by typing an <OK>.

After this command is executed, the statement "Contradictions have been..." is inserted after statement 0, and assigned the statement number 1.

You will be able to see the statements as you enter them into your file. Once an entire screen is filled, however, they will be off the bottom of the screen. You will learn later how to move the statements at the bottom of the screen to the top.

5. You have now learned how to enter one statement; more commonly, you will want to enter several statements, one after the other. Instead of repeating the Insert Statement command for each new statement, type the character <CTRL-E> at the end of your first statement (instead of an <OK>). This tells the system to continue the Insert Statement command. We call this the "enter mode".

Once you get in the "enter mode" by typing the <CTRL-E>, you end each subsequent statement with an <OK>. You can then immediately type in another statement, ignoring the prompt for level adjustment (L:). To exit from "enter mode," follow your last statement with an <OK> and then press the middle button of the mouse <CMD DEL>.

Add three more statements after statement 1. When prompted for the first BUG, place the cursor under any character in statement 1:

```
.....................................................................
You type:
            isBUG<OK>The review meeting will be at
            3:00<CTRL-E>
You see:
            BASE
              > C: Insert > C: Statement (to follow)
              B: BUGMARK T:
              The review meeting will be at 3:00.

You type:
            Only wise, blind men should attend.<OK>
You see:
              L:
              Only wise, blind men should attend.

You type:
            A recurcive redefinition plan should
            emerge.<OK><CMD DEL>

You see:
              L:
              A recurcive redefinition plan should
              emerge.
              L:
              BASE
              > C:
.....................................................................
```

6. You have completed a rough draft of your memo and now should check it for completeness, typing errors, etc. Scan your screen to review the contents of the file.

7. You may now decide that you want to have a new statement 2. When you are working with an entire statement, you may BUG under any character in the statement. To replace statement 2:

```
.....................................................................
You type:
            rsBUGThe final seminar is scheduled for 3:00<OK>
You see:
            BASE
              > C:  Replace > C: Statement (at)
              B: BUGMARK (by) T:
            The final seminar is scheduled for 3:00
.....................................................................
```

8. Now you might decide that statement 3 is superfluous. To delete statement 3, BUG anywhere in the statement.

```
.....................................................
You type:
          dsBUG<OK>
You see:
          BASE
            > C:  Delete > C: Statement (at)
            B: BUGMARK
.....................................................
```

Notice that when you deleted statement 3, the system renumbered the remaining statements and you now have a new statement 3.

9. You may also want to add text to the end of statement 2. To do so, you use a command similar to the Insert Statement command, Insert Text. In this case, you BUG the very last character, the last "0," which indicates to the system that the first new character you type will be inserted immediately after the character you BUGGED. Therefore, in this case, the first character you add should be a space.

```
.....................................................
You type:
          itBUG<SP>in the project room.<OK>
You see:
          BASE
            > C:  Insert > C: Text (at)
            B: BUGMARK in the project room.
.....................................................
```

Notice in this command you used the command word Text instead of Statement; thus the insertion becomes part of an existing statement rather than a new statement.

10. There are some characters, words, and phrases on your screen that you might want to change. You can easily make corrections by using the commands Replace, Insert, or Delete, followed by the command words Character, Word, or Text. When you are working with a word, you can BUG any character in the word; when you want only one character, be sure to BUG that specific character. When you need to specify text that may include more that one word, BUG both the starting point and the ending point of the text. In the example that follows, you will replace the character "c" by an "s" in the word "recurcive." Be sure to BUG directly under the second "c."

```
.....................................................
You type:
          rcBUGs<OK>
You see:
          BASE
            > C:  Replace > C: Character (at)
            > B:  BUGMARK (by) s OK
.....................................................
```

Now you want to replace the word "emerge" with "evolve." For the BUG, place the cursor under any character in the word "emerge."

```
.....................................................
You type:
          rwBUGevolve<OK>
You see:
          BASE
            > C:  Replace > C: Word (at)
            > B: BUGMARK (by)evolve OK
.....................................................
```

Finally, you decide you want to eliminate both the words "scheduled" and "for" from statement 2. In this case, if you BUG the space between the two words, both the words will be deleted at the same time.

```
.....................................................
You type:
          dwBUG<OK>
You see:
          BASE
            > C:  Delete > C: Word (at)
            > B: BUGMARK OK
.....................................................
```

Notice how the spaces between the words are closed up when you delete a word (or words). This is the advantage of using the Delete Word command rather than Delete Text when it is appropriate.

11. You are now going to copy what you see on your screen several times. You will use the Copy Group command: BUG statement 1 (not the origin statement), BUG the last statement on the screen, and then BUG the origin statement. A copy of all the statements on your screen will be made and placed after the origin statement. Do this twice.

```
.....................................................................
You type:
          cgBUGBUGBUG<OK>
You see:
          BASE
            > C:  Copy  > C:  Group (from) > B: BUGMARK
           (through >B: BUGMARK (to follow) > B: BUGMARK
            > L:
.....................................................................
```

12. After you have repeated this three times, notice that some of your statements have disappeared off the bottom of the screen. To move the statement that is at the bottom to the top, you use the Jump command and BUG the bottom statement. This will move the statement that was bugged to the top of the File Window, and if any statements follow it, they will be displayed also.

```
.....................................................................
You type:
          jBUG<OK><OK>
You see:
          BASE
            > C:  Jump (to) > B: BUGMARK
.....................................................................
```

To return to the origin statement (statement 0), use the Jump Origin command followed by two OKs. With the Jump Item command, you can specify a statement number; the specified statement will move to the top of your screen. To do this, you type "ji" for Jump Item, follow this with a statement number, and then two OKs.

13. The memo is finished and you now want to make a fresh version of your file that consolidates all of your changes. When you type the Update File command followed by <OK>, your current file (which is now MEMO) is the one that is updated.

```
.....................................................................
You type:
          uf<OK>
You see:
          BASE
            > C:  Update > C: File > OK:
          < DIRECTORYNAME, MEMO.NLS;2 >
.....................................................................
```

The file name is followed by the extension ".NLS;2." This tells you that MEMO is an NLS file and this is the second

version. Each time you update a file, it increases the version number by one. The feedback that your file is updated will appear in the TTY Window.

14. Your work session is now over and you can leave the system. After you type the <OK>, your screen will go blank before you see the following message from the system.

```
.....................................................
You type:
          <SP>1<OK>
You see:
          BASE
            > C:  Logout > OK:
          TERMINATED JOB #, USER DIRECTORYNAME,
          ACCT ###, TTY # AT DATE TIME USED # in #
.....................................................
```

TNLS EDITING SAMPLE SESSION I

NLS User Training Guide

INTRODUCTION

NLS, or the oNLine System, is the name of the computer system you will be using. Online means you receive immediate feedback about what you have just typed at your terminal.

NLS has facilities to let you do almost everything you need to do with text: compose it; edit it; send it to (and receive it from) other persons; file it in one or more categories; cite and easily obtain documents; search for documents by author and subject; search in documents by word or phrase; and print in practically any format.

This sample session demonstrates the commands used for writing a memo and editing it. This process is explained for TNLS, which is the typewriter version of NLS. You will find it useful to be at a terminal, typing in the commands and text as the sample session describes them.

Although this describes specific functions, we add notes at each step which generalize the operation. Given this sample session as a model, the inexperienced user should be able to perform any of the operations described here and refer to other NLS documentation for more information about the system.

Throughout this sample session we spell out the sequence of keys you strike to make something happen and separately show what will appear on your terminal in response. Keys that do not print, such as carriage return and escape (also called "altmode"), are named inside angle brackets, e.g., <CR> and <ESC>. <SP> represents a space. The control key <CTRL> is used like the shift key. You hold it down while you type the letter that is after the hyphen. The notation for control keys is <CTRL-(some control character)>, for example <CTRL-W>.

Here are some control keys to remember:

<CTRL-X>	aborts commands before you have typed a <CR>.
<CTRL-O>	stops printing.
<CTRL-A>	deletes the character you have just typed.
<CTRL-W>	deletes the word you have just typed.

When you see <CR>, use the return or carriage return on your keyboard.

If you get stuck or confused, typing "?" will show you the next possible alternatives. You then type in one of the alternatives and continue your command.

INSTRUCTIONS

Most users of this sample session will reach NLS through the ARPA Network. For the current connection procedures at your site, see someone knowledgeable in NLS. When you have made your connection you will see:

"TENEX 1.##.## SITENAME 1,##.##" and then an "@".

This is the TENEX ready signal also called a herald. TENEX is a system within the computer that provides access to NLS and other programs. The herald tells you that TENEX is waiting for you to type.

1. To identify yourself to the TENEX system at Office-1:

.....

You type:

log<SP>DIRECTORYNAME<SP>PASSWORD<SP><CR>

You see:

@log DIRECTORYNAME
DIRECTORYNAME
JOB # ON TTY # DATE TIME

.....

If you do not know a DIRECTORYNAME or PASSWORD, ask the person in your organization who usually helps people with NLS, or call (415) 326-6200, extension 3630, between 8 a.m. and 5 p.m. Pacific time.

After the job information has been typed by the system, your login is completed and the TENEX herald "@" will again print at the left margin; it is again your turn to type:

2. To enter the TNLS system:

.....

You type:

nls<CR>

You see:

@nls
BASE C:

.....

If you are using a group directory, you will be asked to type an ident after you type "nls<CR>". Type in the ident (yours, one that you have been given, or one that is associated with this particular directory), following it with a <CR>.

When you enter NLS, "BASE" will print in the left margin. BASE is the herald of NLS's central subsystem.

3. Since you are going to write a memo, you will need an empty file (or work space) in which to put it. You give the file a name, in this case memo, so that you can call it back in future NLS sessions.

```
.....................................................
You type:
          <SP>crfmemo<CR>
You see:
          BASE C:  Create C: File
          T: memo
          < DIRECTORYNAME, MEMO.NLS;1,>
          BASE C:
.....................................................
```

Where NLS expects you to do something, it asks you for a command word by prompting you with a C:, and where it expects you to type in some text, it prompts you with T:.

Note: If you are in the middle of a command and type <CTRL-X>, the command will be aborted and the BASE C: will print out again. Then you may begin again.

The system usually completes the command word for you after you have typed in the first letter. In the case of some commands used less often, you have to type a space and then one, two, or three letters.

You now have a new and empty file named MEMO. Filenames may include up to 29 letters and/or digits, and must begin with a letter. File names may not include spaces, commas, periods, or semi-colons.

If you leave the system without finishing your work on this file (or if you create another file), you can retrieve it (or any other stored file) in TNLS by using the command Jump Link, and typing in the filename, followed by a comma, after the T:. YOU DO NOT NEED TO DO THIS NOW, because you are already in the file named MEMO since you just created it. The command is shown here for your future use:

```
.....................................................
You type:
          jlmemo,<CR>
You see:
          BASE C: Jump C: Link
          T: memo,
          BASE C:
.....................................................
```

4. Now that you have created MEMO, the system has already inserted some information at the file's beginning, i.e., at the statement numbered 0. Statement 0 identifies the file MEMO and is generally unused by you except to describe the beginning of the file. To see the statement you are currently at, i.e., statement 0, type \ at BASE C:

```
.....................................................
The response will be:

BASE C: \
< DIRECTORYNAME, MEMO.NLS;1, >,   DATE   TIME  IDENT ;;;;
BASE C:
.....................................................
```

5. You begin writing your memo by indicating you are going to insert a statement into your file MEMO starting after statement 0, and then by actually typing in some text. Statements are comparable to headings or paragraphs. The system will automatically move the print-head back when it runs out of room at the end of a line. You do not need a carriage return at the end of each line. The lines may not be the same as in the examples. The typogoophical errors are intentional so you can practice some editing later.

```
.....................................................
You type:
          is0<CR><CR>Contradictions have been alledged in
          our description of the elephant.<CR>
You see:
          BASE C: Insert C: Statement (to folow) A: 0
          L:
          T: Contradictions have been alledged in our
          description of the elephant.
          BASE C:
.....................................................
```

Notice that you are prompted for specific types of input. In this case A: asks you for an address, T: for typein. An

address specifies a point in a file. You gave an address of "after statement 0" because that was where you wanted your new statement to begin. If you were creating a file that used an outline structure, L: would prompt you to specify the level in the outline of the new statement being typed in. In this sample session you can ignore L: by typing a <CR>.

After this command is executed, the statement "Contradictions have been..." is inserted after statement 0, and assigned the statement number 1.

6. As you enter statements into the file, you will periodically want to check how the memo looks. You can look at all or part of your file by printing it. To see only the statement where you are, type: \

.....
The response will be:

BASE C: \
1 Contradictions have been alledged in our description of the elephant.
BASE C:
.....

Later on when there are more statements in your file you can see more by using the Print File command, described in step 8.

7. Step 5 showed you how to enter one statement; more commonly, you will want to enter several statements, one after the other. Instead of repeating the Insert Statement command for each new statement, type the character <CTRL-E> at the end of your first statement (instead of a <CR>). This tells the system to continue the Insert Statement command. We call this the "enter mode". Once you get in the "enter mode" by typing the <CTRL-E>, you end each statement typed in with a <CR>, and then (after a <CR> for the L:) type in another statement. Follow your last statement with a <CR> and a <CTRL-X>. The <CTRL-X> takes you out of the "enter mode". To add (after statement 1) three more statements to your file, completing the rough draft of your memo:

```
You type:
          is1<CR><CR>The review meting will be at
          3:00<CTRL-E><CR>
          Only wise, blind men should attend.<CR>
          A recurcive redefinition plan should
          imerge.<CR><CTRL-X>
You see:
          BASE C: Insert C: Statement (to follow) A: 1
          L:
          T: The review meeting will be at 3:00<^E>
          L:
          T: Only wise, blind men should attend.
          L:
          T: A recurcive redefinition plan should imerge
          L:
          BASE C:
```

8. You have now completed a rough draft of your memo and want to check it for completeness, typing errors, etc. To review the content of the file you can use the Print File command. The Print File command starts printing from Statement 0 to the end of the file. Type the following to see the contents of the file you just entered:

```
You type:
          pf<CR>
You see:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MEMO.NLS;1 >, DATE TIME
          IDENT ;;;;
          1 Contradictions have been alledged in our
          description of the elephant.
          2 The review meeting will be at 3:00
          3 Only wise, blind men should attend.
          4 A recurcive redefinition plan should
          imerge.
          BASE C:
```

9. You may now decide that you want to change statement 2. To replace statement 2:

```
You type:
          rs2<CR>The final seminar is scheduled for
          3:00<CR>
You see:
          BASE C: Replace C: Statement (at) A: 2
          T: The final seminar is scheduled for 3:00

          BASE C:
```

10. Now you might decide that statement 3 is superfluous. To delete statement 3:

```
You type:
          ds3<CR><CR>
You see:
          BASE C: Delete C: Statement (at) A: 3 OK:

          BASE C:
```

11. You may also want to add text to the end of statement 2. To do so you use a command similar to the Insert Statement command.

```
You type:
          it2<SP>+e<CR><SP>in the project room.<CR>
You see:
          BASE C: Insert C: Text (to follow) A: 2 +e
          T:  in the project room.
          BASE C:
```

The significant difference in this command from the way you inserted statements is that you specify where in the statement you want the text to go. The "+e" after the statement number tells the system to insert the text at the end of that statement. Notice also that you use the command word Text instead of Statement; thus the insertion becomes part of an existing statement rather than a new statement.

Note also that our example directs you to type a space as the first character of the text you are inserting; that space avoids having "...3:00in the..." appear in the file.

12. If you strike \ after "BASE C:", you can look at the statement you have just edited, statement 2, to check the changes.

.....
The response will be:

2 The final seminar is scheduled for 3:00 in the project room.
BASE C:
.....

13. At this point you are ready to check your file for minor errors. You may print it again as you did in step 8 with the Print File command which prints the whole file, beginning with statement 0:

.....
You type:

pf<CR>

You see:

BASE C: Print C: File OK:
< DIR, MEMO.NLS;1, >, DATE TIME IDENT
1 Contradictions have been alleged in our description of the elephant.
2 The final seminar is scheduled for 3:00 in the project room.
3 A recurcive redefinition plan should imerge.

BASE C:
.....

Note that when you deleted the old statement 3 in step 9, the system renumbered the remaining statements.

14. The most convenient way to correct the kind of typographical errors found in this memo is with the Substitute Text command. This command asks you for the correct text and then the text you want replaced (or substituted for). You may specify only one change or several without repeating the command. Statement 3 contains two misspellings:

```
.....................................................................
You type:
          sts3<CR>sive<CR>cive<CR>n
          eme<CR>ime<CR>y<CR>
You see:
          BASE C: Substitute C: Text (in) C: Statement (at)
          A: 3
          <New TEXT> T: sive
          <Old TEXT> T: cive
          (Finished?) S/Y/N:
          <New TEXT> T: eme
          <Old TEXT> T: ime
          (Finished?) S/Y/N:
          Substitutions made: 2
          BASE C:
.....................................................................
```

Use this command cautiously. You must eliminate ambiguities and avoid causing the system to make substitutions that you don't want. For example, in the first substitution if you had specified "e" for "i" instead of "eme" for "ime", the system would have changed ALL occurrences of the the letter "i". Make the text string unique to avoid surprises.

15. To check statement 3 strike \ when "BASE C:" appears:

```
.....................................................................
The response will be:

BASE C: \
3 A recursive redefinition plan should emerge.
BASE C:
.....................................................................
```

16. The memo is finished and you now want to make a fresh version of your file that consolidates all of your changes.

```
.....................................................................
You type:
          uf<CR>
You see:
          BASE C: Update C: File OK:/C:
          < DIRECTORYNAME, MEMO.NLS;2, >
          BASE C:
.....................................................................
```

17. Your may decide you do not wish to keep the online version of your memo. To delete your file from the online directory:

```
.....................................................................
You type:
          dfmemo<CR><CR>
You see:
          BASE C: Delete C: File T: memo OK:
          Deleted Files are:
          <  DIRECTORYNAME,  MEMO.NLS;2, >
          <  DIRECTORYNAME,  MEMO.NLS;1, >
          BASE C:
.....................................................................
```

18. Your work session is over and you can leave the system:

```
.....................................................................
You type:
          <SP>l<CR>
You see:
          BASE C:  Logout OK:
          TERMINATED JOB #, USER DIRECTORYNAME,
          ACCT ###, TTY # AT DATE TIME USED # in #
.....................................................................
```

TNLS EDITING SAMPLE SESSION II

NLS User Training Guide

INTRODUCTION

"Editing Sample Session II" will be most helpful to you if you work on it after you have completed "Editing Sample Session I". This session introduces you to file structure and illustrates the Copy and Move commands for Statements and STRUCTURES within your file. Your task is to create and revise a multi-leveled outline.

INSTRUCTION

1. You will begin this sample session by creating a file for your outline. You give your file the name "MENU" and are ready to begin inserting statements.

```
.....................................................
You type:
          <SP>crfmenu<CR>
You see:
          BASE C:  Create C: File T: MENU
          < DIRECTORYNAME, MENU.NLS;1, >
.....................................................
```

2. To transcribe an outline you will use Insert Statement and <CTRL-E> "enter mode". This is the same command you used in the "Editing Sample Session I". This time, however, you will be inserting statements at different levels. Thus you must have a clear idea of the structure of your outline and the relationships of the statements in it to one another.

You specify a LEVEL-ADJUST when prompted by L:. The level which you specify is in relationship to the preceding statement. To change the level of a statement you use either "u" (up) to indicate a higher level, or "d" (down) for a lower level. You can type more than one u to go up more than one level. (For example, to insert a statement three levels above the preceding statement, you would type "uuu" at the prompt L:.) You cannot, however, type more than one "d". To insert a statement at the same level as the one preceding it, simply ignore the LEVEL-ADJUST and begin typing your text.

For this sample session you will work with the following outline, given here in its entirety. The first few commands will be illustrated for you; you will then continue on your own until all statements of the outline are inserted in your file. After you have finished inserting the outline you will find a copy of how your file should look. A reproduction of what you see on your terminal as you copy the outline into your file is also provided. We recommend that you only use this to check your work. It should not be depended upon to create the outline.

.....OUTLINE TO BE TRANSCRIBED.....

- Appetizers
  - antipasto
  - asparagus vinaigrette
  - escargot
- Soups
  - Cold Soups
    - vichysoisse
    - gazpacho
      - chopped green peppers, tomato, and cucumber
  - Hot Soups
    - borscht
      - with sour cream
    - chicken
    - spinach
- Salads
  - caesar
  - bean
  - greek
    - with anchovies
  - tossed green
- Entrees
  - Fish and shellfish
    - sole
    - scallops
      - broiled
      - sauteed
    - lobster
      - 1 lb.
      - 2 lb.
  - Meat
    - filet mignon
    - prime rib
- Starch
  - potatoes
    - baked
      - with sour cream and chives
      - with melted cheese
    - french fried
    - whipped
    - creamed
  - rice
    - plain
    - pilaf

.....

3. You begin by inserting the first statement of your outline below the origin statement (statement 0). This will be statement 1 in your outline.

.....
You type:

```
is0<CR>Appetizers<CR>
```

You see:

```
BASE C: Insert C: Statement (to follow) A:  0
L:
T: Appetizers
```

.....

4. The next statement you insert will be a substatement of statement 1. This will become statement 1A. You specify its level by typing a "d" after the prompt L:.

.....
You type:

```
is<CR>d<CR>antipasto<CR>
```

You see:

```
BASE C: Insert C: Statement (to follow) A: L: d
T: antipasto
```

.....

5. The third and fourth statements you will be inserting will also be substatements of statement 1. They will follow statement 1A on the same level, becoming statements 1B and 1C. Use Insert Statement for the third statement, followed by <CTRL-E> to continue in the enter mode. Since both statements are on the same level as statement 1A, ignore the LEVEL-ADJUST and keep on typing. Note that you will no longer be prompted for an address. Statements will automatically follow each other.

.....
You type:

```
is<CR>asparagus vinaigrette<CTRL-E>
escargot<CR>
```

You see:

```
BASE C: Insert C: Statement (to follow) A: L:
T: asparagus vinaigrette<^E>
L:
T: escargot
```

.....

6. You may wish to check your work at this point. Type a <CTRL-X> to take you out of the enter mode, then use the Print File command.

```
.....................................................................
You type:
          <CTRL-X>pf<CR>
You see:
          L:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MENU.NLS;1, >, DATE TIME
          IDENT ;;;;
          1 Appetizers
             1A antipasto
             1B asparagus vinaigrette
             1C escargot
.....................................................................
```

7. Continue to insert statements, adjusting the level of each to the proper level of your outline. Begin again using Insert Statement, followed by <CTRL-E>.

```
.....................................................................
You type:
          is1c<CR>u<CR>Soups<CTRL-E>
You see:
          BASE C: Insert C: Statement (to follow) A: 1c
          L: u
          T: Soups<^E>
.....................................................................
        CONTINUE IN THIS MODE UNTIL THE ENTIRE
    OUTLINE SHOWN ON PAGE 3 IS COPIED INTO YOUR FILE.
.....................................................................
```

8. You should now have the whole outline--from Appetizers to Starch--inserted into your file. To see your file, first type a <CTRL-X> to take you out of the enter mode, then give the command Print File.

```
.....................................................................
You type:
          <CTRL-X>pf<CR>
You see:
          L:
          BASE C: Print C: File OK:
          < DIRECTORYNAME, MENU.NLS;1, >, DATE TIME
          IDENT ;;;;
```

```
1 Appetizers
   1A antipasto
   1B asparagus vinaigrette
   1C escargot
2 Soups
   2A Cold Soups
      2A1 vichysoisse
      2A2 gazpacho
         2A2A chopped green peppers, tomato,
              and cucumber
   2B Hot Soups
      2B1 borscht
         2B1A with sour cream
      2B2 chicken
      2B3 spinach
3 Salads
   3A caesar
   3B bean
   3C greek
      3C1 with anchovies
   3D tossed green
4 Entrees
   4A Fish and shellfish
      4A1 sole
      4A2 scallops
         4A2A broiled
         4A2B sauteed
      4A3 lobster
         4A3A 1 lb.
         4A3B 2 lb.
   4B Meat
      4B1 filet mignon
      4B2 prime rib
5 Starch
   5A potatoes
      5A1 baked
         5A1A with sour cream and chives
         5A1B with melted cheese
      5A2 french fried
      5A3 whipped
      5A4 creamed
   5B rice
      5B1 plain
      5B2 pilaf
```

9. The following is a transcription of what your commands should have looked like as they appeared on your terminal. As stated earlier, use this only to check your work if you find mistakes after you print your file.

.....

COMMAND TRANSCRIPTION

You see:

```
L: d
T: Cold Soups
L: d
T: vichysoisse
L:
T: gazpacho
L: d
T: chopped green peppers, tomato, and cucumber
L: uu
T: Hot Soups
L: d
T: borscht
L: d
T: with sour cream
L: u
T: chicken
L:
T: spinach
L: uu
T: Salads
L: d
T: caesar
L:
T: bean
L:
T: greek
L: d
T: with anchovies
L: u
T: tossed green
L: u
T: Entrees
L: d
T: Fish and shellfish
L: d
T: sole
L:
T: scallops
```

```
L: d
T: broiled
L:
T: sauteed
L: u
T: lobster
L: d
T: 1 lb.
L:
T: 2 lb.
L: uu
T: Meat
L: d
T: filet mignon
L:
T: prime rib
L: uu
T: Starch
L: d
T: potatoes
L: d
T: baked
L: d
T: with sour cream and chives
L:
T: with melted cheese
L: u
T: french fried
L:
T: whipped
L:
T: creamed
L: u
T: rice
L: d
T: plain
L:
T: pilaf
BASE C:
```

.....

10. An inital version of the outline now exists in your file. The remainder of the sample session will illustrate some commands to change your outline. The first task will be to add another statement to the outline. You will use the Insert Statement command to add a substatement to statement 2b2.

```
.....................................................................
You type:
          is2b2<CR>d<CR>with dumplings<CR>
You see:
          BASE: Insert C: Statement (to follow) A: 2b2 L: d
          T: with dumplings
.....................................................................
```

11. You would like statement 5A4 to be in a different location. The Move Statement command allows you to change the location of a statement. The command takes the entire statement and places it where you indicate and at the level you specify relative to a preceding statement. Here, you want the statement to follow 5a. You will want to make this second statement a substatement of the 5a, and will indicate this by typing a "d" after the prompt L:.

```
.....................................................................
You type:
          ms5a4<CR>5a<CR>d<CR>
You see:
          BASE C: Move C: Statement (from) A: 5a4 (to follow)
           A: 5a L: d
.....................................................................
```

You have now rearranged the order of statements in branch 5a. You can see it by using Print Branch command.

```
.....................................................................
You type:
          pb5a<CR><CR>
You see:
          BASE C: Print C: Branch (at) A: 5a V:
             5A potatoes
                5A1 creamed
                5A2 baked
                   5B1A with sour cream and chives
                   5B1B with melted cheese
                5A3 french fried
                5A4 whipped
.....................................................................
```

12. You may want to repeat a statement you have already inserted. The Copy command allows you to make an exact duplicate of a statement and place it where you indicate. You will now copy the statement 2B3 spinach to follow 3C greek. You will indicate that it is to be at the same level as 3C by typing a <CR> when prompted with L:.

```
.....................................................................
You type:
          cs2b3<CR>3c<CR><CR>
You see:
          BASE C: Copy  C: Statement (from) A: 2b3
          (to follow) A: 3c
          L:
.....................................................................
```

13. The next command uses Branch (a statement plus all its substatements, all their substatements, and so on). You will Move Branch 4B (which includes the substatements 4B1 and 4B2) to follow statement 4 so it will become the new branch 4A.

```
.....................................................................
You type:
          mb4b<CR>4<CR>d<CR>
You see:
          BASE C: Move C: Branch (from) A: 4b
          (to follow) A: 4
          L: d
.....................................................................
```

To see the new structure, type Print Branch, using branch 4 as your ADDRESS.

14. You can also duplicate branches that already exist. In this step you will Copy Branch 1 to follow Branch 1.

```
.....................................................................
You type:
          cb1<CR>1<CR><CR>
You see:
          BASE C: Copy C: Branch (from) A: 1
          (to follow) A: 1
.....................................................................
```

15. Now you decide you don't want the branch you just copied so you try the Delete Branch command.

```
.....................................................................
You type:
          db1<CR><CR>
You see:
          BASE C: Delete C: Branch (at) A: 1
          OK:
.....................................................................
```

TNLS EDITING SAMPLE SESSION III

NLS User Training Guide

INTRODUCTION

You will want to work on this sample session after you have completed "Editing Sample Sessions I and II." In this session you create a first draft of a report and revise it into a more polished form. To accomplish this, the sample session shows you how to perform several editing commands which can be applied generally whenever you want to modify text online.

INSTRUCTION

1. To begin this sample session you first need to create a work space. To do so, create a file called "Editing".

.....

You type:

<SP>crfediting<CR>

You see:

BASE C: Create C: File T: editing

<DIRECTORYNAME, EDITING.NLS;1,>

.....

2. You first want to transcribe, or write, a draft of the report online. You will "write" your first draft by using the Insert Statement command and <CTRL-E> to put you into the "enter mode".

.....

You type:

is0<CR>This report will cover a few commands
we've learned thus far, and some new
commands.<CTRL-E>

You see:

BASE C: Insert C: Statement (to follow) A: 0
L:
T: This report will cover a few commands we've
learned thus far, and some new commands.

.....

3. Since you ended your last command with the <CTRL-E>, you can now insert a series of statements without repeating the command words. The level for each statement is indicated in parenthesis before the text. There are some intentional errors in the text which you will be inserting. Try to type in exactly what you see so that you can successfully work through all of the later steps in the sample session. When you are finished inserting statements, you leave the "enter mode" by typing a <CTRL-X> after giving your final <CR>.

The level for each statement is indicated in parenthesis before the text. Notice the different levels for the statements in the report. When you do NOT want to change the level of the statement you are inserting, simply ignore the prompt L: and begin typing in your text.

.....

You type:

Commands already learned:<CR>

You see:

L:
T: Commands already learned:

.....

You type:

d<CR>
Insert: This command allows yu to add, duplicate, or create information in a file. The command Insert Statement was presented in the "Editing Sample Session I." This scenario adds Insert Word and Character.<CR>

You see:

L: d
T: Insert: This command allows yu to add, duplicate, or create information in a file. The command Insert Statement was presented in the "Editing Sample Session I." This scenario adds Insert Word and Character.

.....Continue in this mode,.....
adding the following statements:

(same level)Replace: This command allows you to erase a STRING or STRUCTURE at a specific DESTINATION and put in some other content.

(same level)Delete: Delete erases something that you specify, such as a character or statement, from the DESTINATION you specify.

(same level)Copy: The Copy command is used to reproduce a SOURCE (such as STRING or STRUCTURE) at a specific place.

(same level)Substitute: The command Substitute allows you to put a new STRING in the place of an old STRING everywhere it appears in the STRUCTURE you specify. Substitute is the most common editing command used on the typewriter terminal.

(same level)Move: Move transfers a specific SOURCE (such as STRING or STRUCTURE) to a DESTINATION you specify.

(up a level)New Commands:

(down a level)Transpose: The Transpose command allows you

to make STRINGS or STRUCTURES trade places. This command is introduced in this sample session.

(same level)Append: The Append command attaches one statement to another. The appended statement is added to the end of the receiving statement. You may join the two statements by typing something when prompted for CONTENT. This command is introduced in this sample session.

(same level)Break: The Break command allows you to divide a statement into two. It will break at the next space after the last character you specify in the ADDRESS. You may also specify the level of the second statement relative to the first one.

.....

4. Now that you've written your first draft, you will want to see it. The quickest way to have your whole file printed is with the Print File command.

.....

You type:

```
pf<CR>
```

You see:

```
BASE C: Print C: File OK:
< DIRECTORYNAME, EDITING.NLS;1, >
```

.....

Following the origin statement (shown above), appears the contents of your new file. You can now review it for corrections and additions.

After each of the editing steps, you may want to print the statement you just edited. Type ps<CR><CR> which is the Print Statement command or type \ at BASE C:.

5. The first obvious error is in statement 2a, where "you" is misspelled. The command Insert Character can correct this error. You type the command with the statement number and the letter "y" for your ADDRESS. The character you add will be inserted after the last character you specify in your ADDRESS.

```
.....................................................................
You type:
          ic2a<SP>"y"<CR>o<CR>
You see:
          BASE C: Insert C: Character (to follow) A: 2a "y"
          T: o
.....................................................................
```

You must always be sure that your ADDRESS is unique to insure that the character is inserted in the right place. In this case "y" was sufficient since there were no other y's in the statement.

6. The next revision you want to make is also in statement 2a. To clarify the last sentence, you decide to add the word "Insert" after "and". To do so, use the Insert Word command.

```
.....................................................................
You type:
          iw2a<SP>"<SP>and"<CR>Insert<CR>
You see:
          BASE C: Insert C: Word (to follow) A: 2a " and"
          T: Insert
.....................................................................
```

Again, you needed a unique STRING for your ADDRESS. You typed "<SP>and" to distinguish the STRING from the letters "and" in the word "command".

7. Your next correction is in statement 2c, where you want to delete the unnecessary word "that". You will use the Delete Word command.

```
.....................................................................
You type:
          dw2c<SP>"that"<CR><CR>
You see:
          BASE C: Delete C: Word (at) A: 2c "that"
          OK:
.....................................................................
```

8. In statements 2b, 2d and 2e you want to substitute the word "specified" for "specific". To make all the edits simultaneously, execute the Substitute Word (in) Branch command, using the branch beginning "Commands already learned" (statement 2) as your ADDRESS.

```
.....................................................
You type:
          swb2<CR>specified<CR>specific<CR><CR>
You see:
          BASE C: Substitute C: Word (in) C: Branch (at)
          A:2

          (New WORD) T: specified

          (for all occurrences of old WORD) T: specific
          (Finished?) S/Y/N:
          Substitute in Progress

          Substitutions made: 3
.....................................................
```

A word of caution here. You must be careful when you are using the Substitute command, particularly in potentially large STRUCTURES such as a branch. Since every place the old STRING appears in the specified branch will be changed, be very sure that you want to change all instances.

9. You would like your report to tell which of the command words the user will be introduced to in "Editing Sample Session III." Statements 3a and 3b already include this information, but you have neglected to add it to statement 3c. The most efficient way to do this might be to simply copy some appropriate text from statement 3a or 3b. The last sentence in statement 3b is suitable.

```
.....................................................
You type:
          ct3b<SP>"T.<SP>"<CR>3b<SP>+e<CR>3c<SP>+e<CR>
You see:
          BASE C: Copy C: Text (from) A: 3b "T. "
          (through) A: 3b +e
          (to follow) A: 3c +e
.....................................................
```

Again, you must be careful to use a unique STRING in the ADDRESS. Notice that this command copies text beginning with the last character in the first ADDRESS STRING. By typing "T.<SP>" you copied the two blank spaces at the end of the sentence, providing proper spacing at the DESTINATION.

10. You would like part of statement 3c to be set off as a separate statement. Break Statement allows you to divide a statement into two. It will break at the next space after the

last character you specify in the ADDRESS. You may also indicate the level of the second statement relative to the first one. You will want to make your second statement a substatement of the first, and will indicate this by typing a "d" after the prompt L:

```
.....................................................
You type:
          bs3c"ESS"<CR><CR>d<CR>
You see:
          BASE C: Break C: Statement (at) A: 3c"ESS"
          L: d
.....................................................
```

By breaking statement 3c and putting the new statement on a lower level, you have created a new branch. You can see it by using the Print Branch command.

11. In this step you will reverse the process carried out in the previous step. Using statements 3c1 and 3c, you will attach one statement to another with the Append Statement command. The appended statement is added to the end of the receiving statement, and is joined with what you type when prompted for CONTENT. You will type <SP><SP> when you are prompted for CONTENT, since you do not need to add any additional text for the new statement to make sense.

```
.....................................................
You type:
          as3c1<CR>3c<CR><SP><SP><CR>
You see:
          BASE C: Append C: Statement (at) A: 3c1
          (to) A: 3c
          (join with) T:
.....................................................
```

12. You need to make another structural change. This time you want to switch the order of the statement beginning "Delete" and the statement beginning "Replace." Use the command Transpose for this change. Check your printed copy for the statement numbers before you execute this comand.

```
.....................................................
You type:
          ts2c<CR>2b<CR><CR>
You see:
          BASE C: Transpose C: Statement (at) A: 2c
          (and) A: 2b
          OK:
.....................................................
```

13. You have completed your revisions and would like to print your new version at your terminal. One way of doing this is to Jump to the beginning of your file (the origin statement), and use the Print Rest command.

```
.....................................................
You type:
          jo<CR><CR>
You see:
          BASE C: Jump (to) C: Origin A: V:
.....................................................
```

This command asks for an ADDRESS and allows you to change your viewspecs. You type a <CR> to indicate the ADDRESS of the file you have loaded, and another <CR> to keep the same viewspecs. Now you are ready to print the file.

```
.....................................................
You type:
          pr<CR>
You see:
          BASE C: Print  C: Rest OK:
          <DIRECTORYNAME, EDITING.NLS;1,> , DATE
          TIME IDENT ;;;;
.....................................................
```

This is then followed by the contents of your file.

14. Now that you have completed work on your file, you will want to use the Update File command which creates a "new" version of your file by incorporating into it all of the modifications you have made at this time.

TNLS EDITING SAMPLE SESSION IV

NLS User Training Guide

INTRODUCTION

You will want to work on this sample session after you have completed "TNLS Editing Sample Sessions I, II, III", and "TNLS File Viewing Sample Session." In this sample session you will learn some advanced editing commands, how to address statements by names, the Sort command and the Modify subsystem Substitute command. To best use this sample session, we have already created a file that is good for editing; the first step of the session will show you how to copy this file into your directory.

INSTRUCTION

1. In this step you will create a new file and then copy into it the contents of a file that already exists.

.....

You type:

<SP>crfzapo<CR>

You see:

```
BASE C: Create C: File T: zapo
<  DIRECTORYNAME, ZAPO.NLS;1  >
BASE C:
```

You type:

cbuserguides,vacation,1<CR><CR>

You see:

```
BASE C: Copy C: Branch (from)
A: userguides,vacation,1
(to follow) A:
L:
BASE C:
```

.....

Now you have a file in your directory called ZAPO.NLS that you will be able to use in this session. You may want to use the Print File command to see the file.

2. The first editing command allows you to change characters to upper or lower case in either a STRING or STRUCTURE that you specify. The Force command can capitilize either a single character, an entire word, or a whole branch, depending on what you specify. First try the Force Character command followed by Print Statement so you can see the changes.

.....

You type:

ps1a<CR><CR>

You see:

```
Base C: Print C: Statement A: 1a OK:
1A AOE (All-Out Extravaganzas)
```

You type:

fc1a"x"<CR><CR>

You see:

```
BASE C: Force (Case) C: Character A: 1a"x" OK:
```

You type:

ps1a<CR><CR>

You see:

Base C: Print C: Statement A: 1a OK:
1A AOE (All-Out EXtravaganzas)

.....

You can see that the "X" in Extravaganza was capitalized.

3. Below are a series of commands to show some different ways to use the Force command:

.....

You type:

fw1a1<CR><CR>

You see:

Base C: Force (Case) C: Word A: 1a1 OK:

You type:

fb1b<CR><CR>

You see:

BASE C: Force (Case) C: Branch A: 1b OK:

You type:

pg1a<CR>1b<CR><CR>

You see:

Base C: Print C: Group A: 1a (through) A: 1b
OK:

1A AOE (All-Out Extravaganzas)

1A1 DEFINITION:

1A1A Characterized by ten years of self-denial spent accruing vacation days and saving 10% of annual income.

1A2 Examples:

1A2A Exploring the architectural wonders and culinary triumphs of the major Hilton Hotels worldwide

1A2B An overland safari in Antartica, photographing icebergs seldom seen by man

1B MA (MODERATE ADVENTURES)

1B1 DEFINITION:

1B1A USUALLY LIMITED TO ONE-TWO WEEKS, AND OFTEN CONFINED TO A BUDGET OF ONE-TWO WEEKS SALARY. SOME EXCEPTIONS, FREQUENTLY IN THE FORM OF A CHECK FROM PARENTS OR IN-LAWS TO ENCOURAGE VISITING.

1B2 EXAMPLES:

1B2A A FUN-FILLED WEEK AT THE SHADY REST MOTEL IN WINNAMUCCA, NEV. (GAMBLING PRIVILEGES AND LIVE ENTERTAINMENT AT THE WINNER'S CASINO, COURTESY OF THE WINNAMUCCA WOMENS WEEKLY)

1B2B SHARING COZY, TWO-ROOM COTTAGE AT THE BEACH WITH ANOTHER COUPLE AND THEIR THREE KIDS, TOO

.....

Note that the first word in statement 1A is capitalized as well as the entire branch 1B.

4. The Force Mode command allows you to change the way Force Case works. Force Mode Lower will make all letters lower case when you use the Force Case command, and Force Case First will make the first letter of each word upper case. This change is only in effect for the session of NLS you are in, and the default will be back in effect when you reenter NLS another time. Try the commands below to see how this works.

.....

You type:

fml<CR>

You see:

BASE C: Force (Case) C: Mode C: Lower OK:

You type:

fs1b<CR><CR>

You see:

BASE C: Force (Case) C: Statement A: 1b OK:

You type:

ps1b<CR><CR>

You see:

BASE C: Print C: Statement (at) A: 1b
V:
1B ma (moderate adventures)

.....

5. Now try the same thing with the Force Case First command. You will notice that the first letter of every word will become capitalized.

```
.....................................................................
You type:
          fmf<CR>
You see:
          BASE C: Force (Case) C: Mode C: First (letter
          upper) OK:

You type:
          fs1b<CR><CR>
You see:
          BASE C: Force (Case) C: Statement A: 1a OK:

You type:
          ps1b<CR<CR>>
You see:
          BASE C: Print C: Statement (at) A: 1a
          V:
          1B  Ma (Moderate Adventures)
.....................................................................
```

6. You have learned how to identify statements by Statement Numbers and SIDs. In addition, NLS allows you to "label" statements. These labels are called names and are words that can be used to address a statement. An NLS name can consist of any alphabetic characters, numbers, -, @, or ', so long as it begins with an alphabetic character. Names must be the first thing in a statement. Names may have different delimiters that can be set by the user. To find out what the name delimiters are for a particular statement, use the Show Name (delimiters) command.

```
.....................................................................
You type:
           <SP>shn1a<CR>
You see:
           BASE C: Show C: Name (delimiters for statement
           at)
           A: 1a
           NULL NULL
.....................................................................
```

NULL NULL name delimiters means that any statement beginning with an alphabetic character will have the name made of the first word in that statement. The system has default name delimiters for any file created and they are NULL NULL.

7. You can use statement names as addresses in any command that prompts you A:. In this example, you will replace a word using the statement name as the address.

```
.....................................................................
You type:
          rwexploring<SP>"architectural"<CR>structural<CR>
You see:
          BASE C: Replace C: Word (at) A: 1a2b
          "architectural" (by)
          T: structural
          BASE C:
.....................................................................
```

Note that you did not need to capitalize the word "Exploring" when you used it as an address. When specifying a statement name, the case of the letters does not matter.

8. Sometimes you may wish to name only some of the statements in a file. You can accomplish this by assigning "delimiters" which will surround names in a file or part of a file; their purpose is to indicate which statements have names. Statement names are surrounded by a left delimiter and a right delimiter. You can set the delimiters for individual statements or for the whole file. Use the Set Name (delimiters) command for Branch 0, which will set them for the entire file.

```
.....................................................................
You type:
          <SP>senb0<CR>(<CR>)<CR>
You see:
          BASE C: Set C: Name (delimiters in) C: Branch
          (at)
          A:0
          (left delimiter) T: (
          (right delimiter) T: )
          BASE C:
.....................................................................
```

9. With the delimiters set to "(" and ")" only those statements beginning with some word in parenthesis will be "named." At this point, in your file ZAP0, none of the statements will be named. Change the first word of one of the statements so that it will be named.

```
.....................................................................
You type:
          rw1a<CR>(AOE)<CR>
You see:
          BASE C: Replace C: Word (at) A:1a (by)
          T: (AOE)
          BASE C:
.....................................................................
```

10. Now statement 1A is named. Viewspec D prints statements without their statement names. When viewspec D is turned on, and you Print Branch 0, the words at the beginning of statements that are surrounded by parenthesis will not be printed. In this case the first word of statement 1a will not be printed. Remember the Print Branch command allows you to specify a viewspec.

```
.....................................................................
You type:
          pb0<CR>D<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: D
          <USERGUIDES,VACATION.NLS;2>, ##-MON-7.....
          1 This report will describe three classifications
          vacations (AOE, MA, and PP), defining and setting
          forth examples of each.  It is hoped that this
          document will aid all employees in selecting the
          vacation best suited for their ......
             1A (All-Out Extravaganzas)
                1A1 Definition:....
.....................................................................
```

11. If you have changed your statement name delimiters for different statements and you want to change them all back to your default, use the Reset Name command to do this.

```
.....................................................................
You type:
          <SP>resnb0<CR>
You see:
          BASE C: Reset C: Name (delimiters in) C: Branch
          (at) A: 0
          BASE C:
.....................................................................
```

Any statements you now insert will have the same name delimiters as your origin statement.

12. Sometimes you would like to organize all the statements in the file alphabetically. The Sort command rearranges the statements in a file in alphabetical order depending on the first characters of each statement. First let's create some dummy statements so that we have something to sort. Insert a statement to follow statement 0 and then insert several more down a level from the first one. This way you will create a new Branch 1 that has several substatements you can use for sorting.

.....

You type:

```
is0<CR><CR>Sorting Branch<CTRL-E>
d<CR>malevolent marshmellows<CR>
```

You see:

```
BASE C: Insert C: Statement (to follow) A: 0
L:
T:  Sorting Branch
L:  d
T:  malevolent marshmellows
```

.....

CONTINUE INSERTING THE STATEMENTS BELOW. When prompted for L:, type a <CR> since you want them all to be at the same level. Type a <CTRL-X> after the final <CR> to exit from insert mode.

jumping jujubes

naughty nightmares

galloping giraffees

dancing dunderheads

lazy limericks

frolicking fragments

Use the command Print Branch 1 to see the results of what you just inserted. When prompted for V:, type C so that your statement names will now show.

13. Now you can use the sort command on branch 1. This will rearrange the branch so that it is in alphabetical order.

```
.....................................................................
You type:
          <SP>sob1<CR>
You see:
          BASE C: Sort C: Branch (at) A: 1
          BASE C:
.....................................................................
```

To see the results of the Sort command, use the Print Branch 1 command followed by two <CR>s.

14. You have available several subsytems that can help with editing. One of the best is the Modify subsystem. One command in the Modify subsystem corrects spacing after punctuation marks. It allows you to specify the number of spaces you want after commas, periods, semicolons, and colons. First make some changes to branch 1 so that you have something to edit.

```
.....................................................................
You type:
          rc1a"<SP>"<CR>:<CR>
You see:
          BASE C: Replace C: Character (at) A: 1a" "
          (by) T: :
          BASE C:

You type:
          rc1b"<SP>"<CR>;<CR>
You see:
          BASE C: Replace C: Character (at) A: 1a" "
          (by) T: ;
          BASE C:

You type:
          rcgalloping"<SP>"<CR>.<CR>
You see:
          BASE C: Replace C: Character (at) A: galloping" "
          (by) T: .
          BASE C:

You type:
          rcjumping"<SP>"<CR>,<CR>
You see:
          BASE C: Replace C: Character (at) A: jumping" "
          (by) T: ,
          BASE C:
.....................................................................
```

Note the use of statements names in two of the preceding Replace commands.

To see your changes in punctuation, use the Print Branch 1 command followed by two <CR>s.

15. You have now made changes to the statements in Branch 1 so that they are grammatically incorrect. You wish to adjust the spacing after the punctuation marks. Goto the subsystem Programs, load the subsystem Modify, Goto that subsystem and use the Substitute command.

.....

You type:

```
gp<CR>
```

You see:

```
BASE C: Goto (subsystem) C: Programs OK:
PROG C:
```

You type:

```
lpmodify<CR>
```

You see:

```
PROG C: Load C: Program T: modify
Loading User Program

Don't Execute via RUN PROGRAM Command
Use GOTO SUBSYSTEM Command
Loading User Program

Subsystem MODIFY NOW Available (Attached)
PROG C:
```

You type:

```
gm<CR>
```

You see:

```
PROG C: Goto (subsystem)  Modify OK:
MODI C:
```

TNLS FILE-VIEWING SAMPLE SESSION

NLS User Training Guide

INTRODUCTION

The "File-Viewing Sample Session" illustrates a variety of ways to see and print your files and Journal mail. Some of the most frequently used viewspecs are introduced, and the Print and Jump commands are explored further. (Note: This sample session covers some of the commands taught in the second TNLS course, "Introduction to Structure and Viewing", and adds some alternative ways of doing things. You may also want to refer to "Editing Sample Sessions I and II".) You will find it useful to be at a typewriter terminal, typing in the commands and text as the sample session describes them.

INSTRUCTION

1. This sample session uses a file that is already written. Use the Jump Link command to get the file USERGUIDES, VACATION as your current file.

```
.....................................................
You type:
          jluserguides,vacation,<CR>
You see:
          BASE C: Jump C: Link T: userguides,vacation,

          < USERGUIDES, VACATION.NLS;1, >
.....................................................
```

2. The online system you are using allows you to "view" your files in several different ways. The appearance of your file is controlled by single-letter codes called viewspecs. Some viewspecs are set automatically when you log in. These are called the default viewspecs. To see a list of the viewspecs currently in force, use the Show Viewspecs (status) command.

```
.....................................................
You type:
          <SP>shv<CR>
You see:
          BASE C:  Show C: Viewspecs (status) OK:
          levels: ALL, lines: ALL, hjmpuzACEHJLP
.....................................................
```

You can use the Show Viewspecs command at any time. If you have just logged in, or have been working for a while but have not changed your viewspecs since logging in, you will see exactly what is shown above. These are the default viewspecs. Viewspec w means show all levels and all lines. Some of the other default viewspecs you will be working with in this sample session are listed below with their values:

J - show statement numbers, not SIDs (Statement Identifiers) when viewspec m is on
m - numbers on (SIDs or Statement numbers)
z - no blank lines between statements
H - SIDs or Statement numbers on left side

For more information about the other default viewspecs, see the "TNLS-8 Quick Reference" card or use the Help command.

Notice that some viewspecs are lowercase letters while others are uppercase. Uppercase viewspecs do different things than do lowercase viewspecs.

3. To see the file with the default viewspecs use the Print File command.

```
.....................................................................
You type:
          pf<CR>
You see:
          BASE C: Print C: File OK:
          < USERGUIDES, VACATION.NLS;1, >, DATE TIME
          IDENT ;;;;
.....................................................................
```

The header will be followed by the rest of the file, each statement beginning with a statement number.

4. One way to change your viewspecs is to use the Set Viewspecs command to enter new codes. In this step you will alter the appearance of your file a great deal by setting viewspec x, which shows one line and one level only. Follow the Set Viewspecs command with the Print File command.

```
.....................................................................
You type:
          <SP>sevx<CR>pf<CR>
You see:
          BASE C:  Set C: Viewspecs V: x

          BASE C: Print C: File OK:
          < USERGUIDES, VACATION.NLS;1, >...
          1 This report will describe three classifica...
.....................................................................
```

Since there is only one top level statement in this file, viewspec x shows you only one line in your entire file.

5. There are other ways to change your viewspecs. Some commands include the prompt "V:" to allow you to type in new viewspecs. Print STRUCTURE is one of these commands. This time, use the viewspecs "dt". Notice their effect.

```
.....................................................................
You type:
          pbO<CR>dt<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: dt

          < USERGUIDES, VACATIONS...
          1 This report will describe three...
.....................................................................
```

The two viewspecs d and t have the same effect as viewspec x: viewspec d shows one level only, and t one line only.

6. This step will experiment with different "clipping" viewspecs--viewspecs that cut off lines or levels. Begin with viewspecs b and r (show one level more; show one line more). You can use more than one viewspec b or r to indicate more than one additional line or level. To see how this works, type in the viewspecs "brr" when prompted in the Print Branch command. You will see one more level and two more lines, or a total of two levels and three lines for each statement. (Since statements 1A through 1C are only one line long, they will print out as a single line.)

```
.....................................................................
You type:
          pb<CR>brr<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: brr

          < USERGUIDES, VACATION...
          1 This report will describe three...
          MA, and PP), defining and setting...
          hoped that this document will aid...
             1A AOE (All-Out Extravaganzas)
             1B MA (Moderate Adventures)
             1C PP (Penniless Pilgrimages)
.....................................................................
```

Viewspecs a and q show one level less and one line less.

```
.....................................................................
You type:
          pb0<CR>aq<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: aq

          < USERGUIDES, VACATION...
          1 This report will describe three...
          MA, and PP), defining and setting...
.....................................................................
```

As with viewspecs b and r, you can use more than one viewspec a or q to indicate fewer levels or lines (for example, typing "qq" in this step would have shown two lines less, leaving only one line to be printed).

7. Viewspecs c and t are the last set of clipping viewspecs with which you will experiment. Viewspec c will show all levels, while viewspec t will show first lines only, as illustrated in step 2. This combination of viewspecs provides a quick skeleton view or outline of your file structure.

```
.....................................................................
You type:
          pb0<CR>ct<CR>
You see:
          BASE C: Print C: Branch (at) A: 0
          V: ct

          < USERGUIDES, VACATION...
.....................................................................
```

The header will be followed by the contents of your file with all levels showing, but only one line of each level.

8. You can also change your viewspecs with the Print Group command. A group is a set of branches where you specify the first and last branch to be included. The two branches you specify must be at the same level. In the followng example, you will print what you did in Step 7, except without statement 0 or 1.

```
.....................................................................
You type:
          pg1a<CR>1c<CR>ct<CR>
You see:
          BASE C: Print C: Group (at) A: 1a (through) A: 1c
          V: ct

          1A  AOE (All-Out Extravaganzas...
.....................................................................
```

The header will be followed by the contents of your file with all levels showing, but only one line of each level.

9. You will work with viewspecs that number your statements in this step. As you learned in step 2, the default viewspecs for statement numbering are m, J, and H. This means statement numbers are turned on and will appear on the left side. You will first change both the kind of numbering and the side on which it is printed. Viewspec I causes SIDs rather than statement numbers to print, and viewspec G puts the numbers to the right of the statement.

```
.....................................................................
You type:
          pb1b<CR>IG<CR>
You see:
          BASE C: Print C: Branch (at) A: 1b
          V: IG

          MA (Moderate Adventures)              014
             Definition:                        015
          .....
.....................................................................
```

This is followed by the remainder of the branch, showing SIDs on the right side following each statement. Note that only one line shows for each statement since viewspec t is still in force.

10. You may use SIDs in your address rather than statement numbers. Try Print Statement (which also prompts you with V:) and change back to statement numbers by using viewspec J.

```
.....................................................
You type:
          ps014<CR>J<CR>
You see:
          BASE C: Print C: Statement (at) A: 014
          V: J

          MA (Moderate Adventures)              1B
.....................................................
```

Although you are now seeing statement numbers, each statement can still be identified by its SID. Use Print Statement again, with an SID address, and move the statement number back to the left side using viewspec H.

```
.....................................................
You type:
          ps015<CR>H<CR>
You see:
          BASE C: Print C: Statement (at) A: 015
          V: H

             1B1 Definition
.....................................................
```

11. Another way of printing a statement is by using the Linefeed command, which prints the next statement. You execute this command by typing the Linefeed key on your terminal or <CTRL-J>. The notation for the Linefeed key is <LF>. In this step, turn on viewspec s so that you can see all lines, then print the statement following statement 015.

```
.....................................................
You type:
          <SP>sevs<CR><LF>
You see:
          BASE C:  Set C: Viewspecs V: s
          BASE C:

               1B1A Usually limited to one-two...
               a budget of one-two weeks salary....
               frequently in the form of a check ...
               encourage visiting.
.....................................................
```

12. The command Reset Viewspecs changes the viewspecs back to your default set for the session in which you are working. Typing

Reset Viewspecs now will bring the default viewspecs back into force.

.....

You type:

<SP>resv<CR>

You see:

BASE C: Reset C: Viewspecs OK:

.....

If you would now like to see your viewspecs, use the Show Viewspecs command. The list that prints out will be identical to that which you saw in step 2.

13. You will use a viewspec with which you are already familiar to begin this step.

.....

You type:

pb1c<CR>x<CR>

You see:

BASE C: Print C: Branch (at) A: 1c
V: x

1C PP (Penniless Pilgrimages)

.....

To change the effects of viewspec x back to all lines and all levels, use the default viewspec w. Viewspec n will turn off the statement numbers. Also try viewspec y, which turns on blank lines between statements.

.....

You type:

pb1c<CR>wny<CR>

You see:

BASE C: Print C: Branch (at) A: 1c
V: wny

.....

Branch 1C will follow, with all lines and levels showing and blank lines between the statements. Now try the default viewspec z, which turns off lines between statements.

```
.....................................................................
You type:
          pb1c<CR>z<CR>
You see:
          BASE C: Branch (at) A: 1c
          V: z
.....................................................................
```

You will now see the same Branch 1C without blank lines between statements.

14. The rest of this sample session will explore some ways to move among files. First try the Jump to File Return command. This will take you back to the file that was current before the VACATION file. If you had just logged in, you will be back at your initial file.

```
.....................................................................
You type:
          jfr<CR><CR>
You see:
          BASE C: Jump (to) C: File C: Return OK:
          "< DIRECTORY, FILENAME >" Y/N: OK:
          < DIRECTORY, FILENAME >
.....................................................................
```

Notice that you are prompted by Y/N in the command. This gives you the option to return to other files you have been working on in this work session. If you had typed "n" rather than a <CR> (or "y"), the name of the file you had been working in previous to the one you have now returned to would have appeared, and so on.

15. There is another way to go from your current file back to the previous one. You can use the Print STRUCTURE command and give ".fr" (file return) for your address. Try it here with the Print Statement command. This will take you back to statement 1C in USERGUIDES, VACATION, since that's where you were last working in that file.

```
.....................................................
You type:
          ps.fr<CR><CR>
You see:
          BASE C: Print C: Statement (at) A: .fr
          V:

             PP (Penniless Pilgrimages)
.....................................................
```

Note that you were given the opportunity to change your viewspecs in this command. By typing a <CR> you left them as they were when you were last in that file.

16. The Print Branch command also allows you to move to another file when your address specifies it. In this step you will go to your initial file and print your first Journal citation. (If you have not yet received any Journal mail, come back to complete this sample session when you have. If you do not read your Journal mail, you can stop here.)

For your address, use the FILENAME (IDENT,) followed by the word "journal", and the notation ".n" to indicate the next branch. This will print the first citation in your Journal. Set viewspec m to see the statement number.

```
.....................................................
You type:
          pbIDENT,journal<SP>.n<CR>m<CR>
You see:
          BASE C: Print C: Branch A: IDENT,journal .n
          V: m

            SGR 3-FEB-77 14:32  28744
            Report from an Important Meeting
            Location:  (FJournal, 28744, 1:w)
            *****Note:  [ACTION]*****
.....................................................
```

Your journal citation will not look like the one shown here, but the two should be very similar.

First you will see the IDENT of the person(s) who authored the item. Next is the date and time the item was sent, followed by the unique catalog number assigned to this item. The title of the item is next, followed by the address of the location of the whole document. A link to this address is included--this is explained in the next step.

17. This step will explore the Jump Link command that is useful when you want to read your mail. A link is a string of characters in a statement that names the address of any location in any NLS file (optionally with any viewspecs). Links are surrounded by delimiters: parentheses () or angle-brackets <>. The link you will be using here is in your journal citation, shown above. Notice that there is a string of characters in parentheses following the word "Location:". This is the link to the journal item. When typing in the link, spaces are optional.
The command below causes the file with the journal item to become your current file. You cite the location of the link (in this case 1A), followed by ".1". This indicates you want to jump to the link shown in that statement.

.....
You type:

ja1a.1<CR>

You see:

BASE C: Jump (to) C: Link A: 1a.1

< DIRECTORY, FILE >
.....

To see this journal item you can use the Print File command, unless the journal item is less than 1000 characters. (If this is the case, the item will already have appeared when you printed the branch in your initial file.) You may do this now if you wish. Then return to your initial file with the Jump File Return command (see step 11), and practice a faster way of using links with the Jump to Address command.

You may log out at this point, Jump File Return back to your initial file, or Jump Link to another file.

SAMPLE SESSION SUMMARY

Viewspecs

Single letter codes that control the appearance or "view" of your files. When viewspecs are allowed in a command, you are prompted by a "V:". You may type a string of any of the viewspec codes, terminated by OK. Type just an OK if you don't want to change the viewspecs. Uppercase viewspecs do different things than lowercase viewspecs.

Viewspecs used in this sample session:

x - show one line and one level only
w - show all lines and all levels (default)
c - show all levels
d - show first level only
s - show all lines
t - show first lines only
a - show one level less
b - show one level more
r - show one line more
q - show one line less
m - statement numbers/SIDs on (default)
n - statement numbers/SIDs off
I - show SIDs not statement numbers
J - show statement numbers, not SIDs (default)
G - statement numbers/SIDs right
H - statement numbers/SIDs left (default)
y - blank lines between statement on
z - blank lines between statements off (default)

Show Viewspecs

Lists the viewspecs in force in the current work session.

Set Viewspecs

Allows you to change the viewspecs at anytime for the current work session.

Reset Viewspecs

Sets the viewspecs back to your initial set for the current work session.

Print Statement

Prints at your terminal the statement you specify when prompted for an ADDRESS.

Jump File Return

Moves you to a file where you were before. ("<PAST FILEADDRESS>") is the name of the file you will go to if you answer Yes or type <CR>. If you answer No, the FILEADDRESS before that will appear.

Link

A string of characters in a statement that names the ADDRESS of any location in any NLS file (optionally with any view). Links are surrounded by delimiters in the order and format: < ADDRESS : VIEWSPECS >. Spaces are optional. Delimiters may be parentheses () or angle-brackets <>.

In-file-address Elements

A character preceded by a period. Addresses a position in and among files. Moves you in relation to your current location in the direction that corresponds to the character you type.

.fr - file return
.n - next
.l - link

SENDMAIL SAMPLE SESSION I

NLS User Training Guide

INTRODUCTION

This sample session shows you how to use a subsystem called Sendmail to send messages and documents to people known to the system, and have these messages cataloged and stored in the Journal. The process is explained for a typewriter terminal. You will find it useful to be at a terminal, typing in the commands and text as they are described.

INSTRUCTION

1. This short Sendmail session will teach you to send a message to people recognized by the system. Although Sendmail has a very extensive system for sending, distributing, cataloging, indexing, and storing documents, most of these steps are done automatically (and invisibly) for you.

You first go to the Sendmail subsystem.

```
.....................................................................
You type:
          gs<CR>
You see:
          BASE C: Goto (subsystem) C: Sendmail
          OK:
          SEND C:
.....................................................................
```

Notice that the herald has changed from BASE to SEND, indicating a new subsystem.

2. You will use the Sendmail command Interrogate to send your message. This command automatically begins six different commands for you. The responses you enter have the same effect as executing six commonly used Sendmail commands.

Six questions, each appearing one at a time, will be asked. They are:

- distribute for action to:
- distribute for information only to:
- title:
- type of source:
- show status?
- send the mail now?

You may type in the responses given in this sample session, or use this opportunity to send a Sendmail item of your own.

The Interrogate command will always prompt in the same way. If you do not want to respond to one of the prompts, just type a <CR> (e.g., if you did not want to distribute your message to anyone for information only, you would type a <CR> when prompted).

When you are asked the distribute questions, you should type in the ident(s) of the person(s) you want the journal item to go to. An ident is a unique string of characters that identifies a person and that is associated with that person for Journal delivery.

You have several options when prompted by "type of source:". You will choose Message in this step, a command which allows you to type in one statement (up to 2000 characters). You may edit the message with <CTRL-A> and <CTRL-W> while typing it in. Other "types of sources" are:

Branch
File
Group
Offline
Plex
Statement

For more information about these alternatives, use the Help command.

The "show status" prompt displays to you what the system knows about the Journal item you are in the process of sending. You are shown the questions you have answered and the responses you have given.

```
.....................................................
You type:
          i<CR>
You see:
          Send C: Interrogate OK:
          (distribute for action to:) T:

You type:
          pka<CR>
You see:
          (distribute for action to:) T: pka
          (distribute for information-only to:) T:
```

```
You type:
            paw2<CR>
You see:
            (distribute for information-only to:) T: paw2
            (title:) T:

You type:
            Using the Interrogate Command<CR>
You see:
            (title:) T: Using the Interrogate Command
            (type of source:) C:

You type:
            mThis message is being sent to you so I
            can practice using the Sendmail Interrogate
            command.<CR>
You see:
            (type of source:) C: Message T: This message
            is being sent to you so I can practice using
            the Sendmail Interrogate command.
            (show status?) Y/N:

You type:
            <CR>
You see:
            TITLE: Using the Interrogate Command
            AUTHOR(S):  IDENT
            DISTRIBUTE FOR ACTION TO: pka
            DISTRIBUTE FOR INFO-ONLY TO: paw2

            MESSAGE: This message is being sent to you so
            I can practice using the Sendmail Interrogate
            command.

You type:
            <CR><CR>
You see:
            (send the mail now?) Y/N:
            Completed
```

.....

As you might have recognized, each of the "prompts" in the Interrogate command has as its counterpart an individual Sendmail command. You will get more practice using the individual Sendmail commands in "Sendmail Sample Session II".

3. When you have completed sending your message, return to BASE. To do so, use the Quit command. (You may log out after quitting by typing <SP>l<CR>.)

```
.....................................................................
You type:
          q<CR>
You see:
          SEND C: Quit
          OK:
          BASE C:
.....................................................................
```

4. The place where you receive Sendmail items other people have sent you is in your initial file, under a statement which begins "(Journal)". Your initial file automatically becomes your current file when you enter NLS. When you are in your initial file, you can read your Journal mail with the Print Journal command. (If your initial file is not your current file, type jl INITIALFILENAME,<CR> before you do the following.)

```
.....................................................................
You type:
          pj<CR>
You see:
          BASE C: Print C: Journal (mail)
          OK:
          BASE C:
.....................................................................
```

SAMPLE SESSION SUMMARY

Goto Sendmail:

Takes you to the Sendmail subsystem. You can leave the subsystem by typing Quit.

Interrogate:

Asks you six questions to aid you in sending Journal mail. The responses you enter have the same effect as executing six commonly used Sendmail commands. The questions are:

distribute for action to:
distribute for information-only to:
title:
type of source:
show status?
send the mail now?

SENDMAIL SAMPLE SESSION II

NLS User Training Guide

INTRODUCTION

This sample session gives you more practice using the Sendmail subsystem to send online items to other people known by the system. In the "Sendmail Sample Session I" you used the command Interrogate, which automatically prompts you with several questions to aid you in using Sendmail. Now you will learn to use several new commands without the aid of Interrogate prompts.

INSTRUCTION

1. This sample session first shows you how to send a file through the Sendmail system using individual commands rather than the Interrogate command. For the first part, use any file you chose that is available to you. The instructions below will send whatever file is your current file. If you have a file you wish to use, go on to Step 2. Otherwise, follow the instructions below to create a file to use.

You will create a new file and then copy into it the contents of a file that already exists.

.....

You type:

```
<SP>crfblapo<CR>
```

You see:

```
BASE C: Create C: File T: blapo
<  DIRECTORYNAME,  BLAPO.NLS;1  >
BASE C:
```

You type:

```
cbuserguides,vacation,1<CR><CR>
```

You see:

```
BASE C: Copy C: Branch (from) A:
userguides,vacation,1
(to follow) A:
L:
BASE C:
```

.....

Now you have a file called BLAPO.NLS in your directory that you will be able to use in this session. You may want to use the Print File command to see the file.

2. Make sure that whatever file you want to send through the mail is your current file. Follow the scenario to go to Sendmail, then you give your Sendmail item the title Joyful Jaunts (or any other appropriate title) and indicate to whom you want it distributed and for what reasons. Use the Comment command to add a note about your Sendmail item.

```
.....................................................................
You type:
          gs<CR>
You see:
          BASE C: Goto (subsystem) C: Sendmail
          OK:

You type:
          f<CR>
You see:
          SEND C: File A:

You type:
          tJoyful Jaunts<CR>
You see:
          SEND C: Title T: Joyful Jaunts


You type:
          cVery Interesting!<CR>
You see:
          SEND C: Comment T: Very Interesting!

You type:
          dabev [or some other IDENT]<CR>
You see:
          SEND C: Distribute (for) C: Action (to) T: bev

You type:
          didvn [or some other IDENT]<CR>
You see:
          SEND C: Distribute(for) C: Information (only)(to)
          T: dvn

You type:
          s<CR>
You see:
          SEND C: Send (the mail) OK:
          Completed
.....................................................................
```

Sometimes you may only want to send part of a file as a Sendmail item. You can send either a single statement, a branch, a group, or a plex. The commands for sending these STRUCTURES are as follows:

<SP>s for Statement
b for Branch

g for Group
<SP>pl for Plex

To name a recipient, you type in his/her IDENT, which is a string of characters that identifies that person to Sendmail. This distribution list may contain more than one IDENT, each separated by spaces or commas. A copy of your Sendmail item is automatically delivered to the author (you in this case). It is delivered to your initial file in a branch named "Author". It will look just like any other journal item except that it will say AUTHOR COPY.

3. You may occasionally need more information about someone to whom you wish to send a piece of mail. For example, you may know the last name of your recipient, but not his/her IDENT. The command Show Record followed by a period "." and the person's last name will supply the IDENT and other pertinent information.

.....
You type:
<SP>shr.boli<CR><CR><CR>
You see:
SEND C: Show C: Record (for ident) T: .boli

The following individuals with last name
Boli are already defined
Boli, Beverly, Organization: SRI-ARC,
Ident = BEV

Is this the correct Boli?
Ident BEV accepted

Ident: BEV
Name: Boli, Beverly
Organization: SRI-ARC
Groups: ...
Mail Addresses: ...
Phone: ...
Delivery: ...
.....

Note: If the system has IDENTS for more than one person with the same last name, all the names will be listed with their corresponding idents. You can then type the correct IDENT for for an online and hardcopy address or use the (CTRL-X) to return to SEND C:.

If you had known your recipient's IDENT you could have typed that in rather than the last name. Then the same information would have appeared.

If you only know the beginning of a person's last name, use the Show Record command, type a period ".", the first three letters and then three more periods "...". This will also supply you with IDENT information.

You may search for the IDENT of an individual at any point where you would type in an IDENT in a Sendmail command (for example, in the Distribute for Action command). You follow the same procedure as shown above: type a period, followed by the last name and a <CR>. In a command other than Show Record, the system will understand that this <CR> is only to mark the end of the last name and will not carry out the command.

4. This step will illustrate the commands Message, Show Status (both introduced in "Sendmail Sample Session I" as part of the Interrogate command), and Private. Sometimes you may limit the number of people who have access to a particular journal item. With the Private command, only those people on the distribution list will be able to read the item.

.....

You type:

dabev<CR>

You see:

SEND C: Distribute (for) C: Action (to) T: bev

You type:

dipooh<CR>

You see:

SEND C: Distribute (for) Information (Only) (to)
T: pooh

You type:

<SP>pr<CR>

You see:

SEND C: Private OK:

```
You type:
          mGood morning.  I'm practicing using some
          Sendmail commands.  Hope you have a
          pleasant day.
You see:
          SEND C: Message T: Good morning.  I'm practicing
          using some Sendmail commands.  Hope you
          have a pleasant day.

You type:
          <SP>Shs<CR>
You see:
          SEND C:  Show C: Status OK:
          AUTHOR(S): IDENT(S)pka
          DISTRIBUTE FOR ACTION TO: bev
          DISTRIBUTE FOR INFOR-ONLY TO: pooh

          MESSAGE: Good morning.  I'm practicing using
          some Sendmail commands.  Hope you have a
          pleasant day.

You type:
          s<CR>
You see:
          SEND C: Send (the mail) OK:
          Completed
          SEND C:
```

Return to the Base subsystem by typing Quit <CR>.

SAMPLE SESSION SUMMARY

Goto Sendmail

Takes you to the Sendmail subsystem. You can leave the subsystem by typing Quit.

Statement

Sends a statement in a file. If you type <CR> after Statement, the statement at your current location will be sent.

Branch

Sends a branch in a file. If you type a <CR> after Branch, the branch that begins at your current location is sent.

Group

Sends a group in a file.

Plex

Sends a plex in a file.

File

Sends a file. If you type a <CR> after File, the file you have loaded is sent.

Title

Lets you give your item a title.

Distribute for Action

Asks you to specify the recipient(s) whom you wish to receive the item, and from whom you want some action.

Distribute for Information

Asks you to specify recipients whom you wish to receive the item for information purposes.

Comment

Allows you to add a comment on the item you are sending.

Show Record

Takes an IDENT, last name (preceded by a period), or the first three letters of a last name preceded by one period and followed by three periods, and displays current information from the Identfile about how to contact that person.

Show Status

Displays to you what information you have entered for the journal item you are working on. You are shown the questions you have answered and the responses you have given.

Message

Allows you to type one statement (up to 2000 characters). Use <CTRL-A> and <CTRL-W> to edit and <CR> to terminate the message.

Private

Allows you to limit the number of people who have access to the Sendmail item to the people who are on the distribution list.

DOCUMENT FORMATTING SAMPLE SESSION

NLS User Training Guide

INTRODUCTION

Before starting this sample session, we recommend that you complete "TNLS Editing Sample Session I," "TNLS Editing Sample Session II," and "TNLS File Viewing Sample Session." In this sample session you will learn to use Output Processor directives to create a rough draft and a final format for printing on a typewriter terminal or a line printer. Follow the step by step instructions to format one of your own files. Each step is followed by an example of the commands that you will use and general comments about directives.

Here are some definitions to remember:

PRINT STATEMENT COMMAND--Prints at your terminal the statement you specify when prompted for an address and moves your location to that statement.

PRINT FILE COMMAND--Prints at your terminal the entire file you have loaded without affecting your current location.

VIEWSPECS--Single letter codes that control the appearance or "view" of your files. When viewspecs are allowed in a command, you are prompted by a "V:". You may type a string of any of the viewspec codes, terminated by <CR>. Type just an <CR> if you don't want to change the viewspecs.

SET VIEWSPECS COMMAND--Allows you to change the viewspecs at any time for the current work session.

THE OUTPUT PROCESSOR

The Output Processor is a program that formats an NLS file for printing. When you print a file using the commands Output Terminal or Output Printer, the Output Processor creates an automatic default format that is generally useful, i.e., it breaks the text into pages, numbers the pages, sets up default margins, etc. However, in many cases you may want to alter this format by inserting Output Processor directives. Output Processor directives are instructions used to control the printed appearance of an individual piece of text, a statement, a page, or an entire document. These directives are a series of characters that are easily recognized; in this sample session they will always begin with a period followed by a capital letter and will end with a semicolon. Directives are followed as instructions and do not appear as text when you print your file using the commands Output Terminal or Output Printer. They are ignored as instructions and printed as text when you print a file with the commands Print File or Output Quickprint. For further information on printing files with or without directives see the Printing Commands table at the end of this sample session.

Two examples of useful formats are included in this session: a double-spaced, first draft format and a single-spaced, final format. Please note that the directives you will use to create these formats can be used in many other combinations to produce desired effects.

For each of the two sample formats, the directives have been divided into two groups: those that have a continuous effect, generally controlling the format of the document as a whole, and those that have a one-time effect, formatting an individual page, statement, or piece of text. We recommend that you begin with the simpler task of creating the rough draft format before tackling the more complex final format.

As you learned in the "File Viewing Sample Session," limited control of format is available through the use of viewspecs. We recommend using directives rather than viewspecs for formatting documents; directives are permanent instructions in the file itself and there are directives you can insert that specifically override a user's viewspecs. When you begin to use directives, you will see that they provide a great many more capabilities to control formatting. However, even when you print a file using directives, a few commonly used viewspecs (e.g., those that control line and level truncation) will continue to affect your file. Throughout this sample session we will suggest viewspecs to

set when printing a file to ensure that the appearance of your report is exactly as you planned it.

LEARNING ABOUT DIRECTIVES

This sample session will guide you step by step in creating a rough draft format and a final format. We will list and define the directives you will use to create these formats; however, you may want further information on these or other directives.

For offline information about individual directives you may refer to ARC's Output Processor User's Guide which alphabetically lists and defines Output Processor directives. Particularly useful is the page diagram that portrays the directives that control each specific element of page layout.

For online information about directives and access to lists of directives grouped by function or alphabetcally, type "h <CR>" to reach the Help subsystem and then type "directives <CR>". You can also use the Help subsystem to find out more about individual directives. Type "h <CR>" to reach Help; then type the word "directive<SP>" followed by the directive itself (without the beginning period and ending semicolon), and finish wth a carriage return. Information about the specific directive will be printed out at your terminal. Use (CTRL-X) to get back to BASE C:. Here is an example using the directive ".Ybs;":

```
.....................................................
You type:
          h<CR>directive<sp>Ybs<CR>
You see:
          BASE C: Help OK/T/[A]: T/[A]: directive Ybs
.....................................................
```

CREATING A ROUGH DRAFT FORMAT

1. Begin by choosing a file to format. Select a well-structured file with more than one branch that is more than one page long. For "filename" below, type the name of your chosen file. Update your file before beginning; after you have inserted the directives and printed your rough draft, use the Delete Modifications command to erase all the changes you have made since the start of the sample session.

```
.....................................................
You type:
          jlFILENAME,<CR>uf<CR>
You see:
          BASE C: Jump (to) C: Link T/[A]: FILENAME,
          BASE C: Update C: File OK/C:
.....................................................
```

2. We suggest that you print out a copy of the file with the default format to help you plan your modifications. Set viewspecs "mywGJ" to show all lines, all levels, statement numbers on the right, and blank lines between statements and then print your file through the Output Processor as follows.

```
.....................................................
You type:
          <sp>sevmwyGJ<CR>ot<CR>nnny
You see:
          BASE C: Set C: Viewspecs V: mwyGJ
          BASE C: Output (to) C: Terminal OK/C: OK:
          (Send Form Feeds?) Y/N: (Simulate?) Y/N:
          (Wait at page break?) Y/N:
          (Go?) Y/N:
          Processing Output
.....................................................
```

3. We will now list and define the directives that you will use to create a rough draft format. The rough draft will be suitable for offline editing and proofreading as well as for further online editing sessions. It will have the author and title of the document, one blank line between lines and two blank lines between statements, the first lines of each statement indented five spaces, SIDs on the right, and a header that indicates the draft status of the document.

The following is a list of the directives that will have a continuous effect on the format of the document as a whole:

Vertical spacing

.Ybs=2;
Sets the distance between statements to two spaces.

.Ybl=1;
Sets the distance between lines within a statement to one space.

Header

.H="DRAFT";
The text within quotes will be printed at the top of every page as the header, starting on the page after the one on which this directive appears.

.Hp=FR;
Sets the horizontal position of the header to flush right.

Footer

.Fp=FR;
Sets the horizontal position of the footer to flush right.

Note that in the Output Processor's default format the "footer" is the page number that prints at the bottom of each page. With directives, you can change the text of the footer, prevent it from being printed, or, using the above directive .Fp= ;, change the position of the footer.

Indenting

.Ifirst=5;
The first line of every statement will be indented five spaces.

Numbering

.Sn=Off;
Left statement numbers will not be printed.

.Snf=72;
The right margin for right statement numbers will be 72 characters, counted from the default left margin.

.Snftype=1;
Right statement numbers will be expressed as SIDs when this directive is combined with .Snf=72;.

Since many directives take effect after the line, statement, or page in which they appear in the file, the directives that you

will use to control the format of the document as a whole are inserted in the beginning of the file in the origin statement. This is to ensure that the directives will take effect at the beginning of statement 1. These directives will control the format of the entire file unless you change them or turn them off by inserting new directives at some point in your file.

Note that many directives will include a number or some characters after the equal sign. This is referred to as the directive "argument" and is a variable--that is, you can specify whatever value you choose. In the case of the directive .Ifirst=5;, we have specified the number five for the argument so that the first line of every statement will be indented FIVE characters. If a value is not specified for the argument, the default is one, e.g., .Gcr; means generate ONE carriage return. The letter "n" is used to indicate that you can insert a specific value for the argument: .Ifirst=n; means that you can specify any number for the number of spaces to be indented.

4. Use the command Insert Text to insert the directives listed above to follow the four semicolons in your origin statement. Type a space between each directive; this is not necessary but it is helpful for editing and proofreading.

```
.....................................................................
You type:
          it0+e<CR> .Ybs=2; .Ybl=1; .H="DRAFT"; .Hp=FR;
          .Fp=FR; .Ifirst=5; .Sn=Off; .Snf=72; .Snftype=1;<CR>
You see:
          BASE C: Insert C: Text (to follow) A: 0+e
          T/[A]:   .Ybs=2; .Ybl=1; .H="DRAFT"; .Hp=FR; .Fp=FR;
          .Ifirst=5; .Sn=Off; .Snf=72; .Snftype=1;
.....................................................................
```

5. Use the command Print Statement to check for typing errors.

```
.....................................................
You type:
          ps0<CR><CR>
You should see:
          BASE C: Print C: Statement (at) A: 0
          V:

          < DIRECTORY NAME, FILE NAME.NLS;2, >, DATE TIME
          IDENT ;;;; .Ybs=2; .Ybl=1; .H="DRAFT"; .Hp=FR;
          .Fp=FR; .Ifirst=5; .Sn=Off; .Snf=72; .Snftype=1;
.....................................................
```

6. Proofread the directives carefully. A directive that is not preceded by a period and ended with a semicolon or one that contains mistyped characters will not be followed as a directive and will be printed as ordinary text. Make any necessary edits to correct errors.

7. The following is a list of directives that have a one-time effect and that you will use to create a title, separate chapters, and other special formatting.

Horizontal spacing

.Center;
: Center the line that includes this directive.

.Center=2;
: Center two lines, starting with the line that includes this directive.

Vertical spacing

.Gcr;
: Generate one carriage return.

.Gcr=2;
: Generate two carriage returns.

Pagination

.Pes;
: Begin a new page at the end of this statement.

.Pbs;
: Begin a new page before the statement that includes this directive.

.Pn=0;
: Set the current page number to zero.

8. To create the title, insert a statement to follow statement 0 that contains the title of the document and the author(s). There will be two lines in the title, one for the title of the paper and one with the name of the author or authors. The directive .Center=2; to centers both lines. Use the directive .Gcr; like a carriage return on a typewriter to break the line after the title and begin the new line with the author's name. To set off the first statement from the body of the text, use .Gcr=2; following the author to add an extra space between the first and second statements. Note: Type the title in all capital letters or simply capitalize the first letters only.

```
.....................................................
You type:
          is0<cr><cr>.Center=2;TITLE.Gcr;Author(s).Gcr=2;
You see:
          BASE C: Insert C: Statement (to follow) A: 0
          L:
          T/[A]: .Center=2;TITLE.Gcr;Author(s).Gcr=2;
.....................................................
```

9. Use the command Print Statement to check for typographical errors. Make any necessary corrections.

10. To create separate chapters, use the directive .Pbs;. (Since most documents are structured so the chapter or section headings are top-level statements, for the purposes of this sample session you may want to consider your top level statements as chapter headings.) Insert the directive .Pbs; at the end of top-level statements to ensure that section headings begin on a new page. Since you have added a top-level statement to your file for the title, you will not need to begin a new page for a chapter or section until statement 3 (as in the following example) or whatever statement is the second heading after your title. Here is an illustration.

```
.....................................................
You type:
          it3+e<CR>.Pbs;<CR>
You see:
          BASE C: Insert C: Text (to follow) A:3+e
          T/[A]: .Pbs;
.....................................................
```

11. Add .Pbs; to follow all the top-level statements or section headings in your file.

12. Use the command Print Statement to check for typographical errors. Make any necessary corrections.

13. To print the origin statement of your file on a separate "throw away" page, put the directive .Pes; in the origin statement. This forces the statement that follows the origin statement to be on a new page. Use the directive Pn to compensate for this additional page. Setting the page numbering to start at 0 means the origin statement will appear on page 0 (the throw away page), and the document will begin appropriately on page 1.

```
.....................................................
You type:
          it0+e<CR> .Pn=0; .Pes;<CR>
You see:
          BASE C: Insert C: Text (to follow) A: 0+e
          T/[A]:  .Pn=0; .Pes;
.....................................................
```

14. Use the command Print Statement to check for typographical errors. If your file is short, you may want to use the Print File command to print out your entire file at your terminal to check all of the directives. Make any necessary corrections.

15. Now you are ready to print out a copy of the rough draft of your document. Make sure to set your viewspecs to show all lines and levels. Use the command Output to Terminal. (If you have access to a line printer, see the chart at statement 6 for the appropriate command.)

```
.....................................................
You type:
          <SP>sevw<CR>ot<CR>nnny
You see:
          BASE C: Set C: Viewspecs V: w
          BASE C: Output (to) C: Terminal OK/C: OK:
          (Send Form Feeds?) Y/N: (Simulate?) Y/N:
          (Wait at page break?) Y/N:
          (Go?) Y/N:
          Processing Output
.....................................................
```

16. You may now either choose to incorporate the directives into your file by using the Update File command or remove them from the file by using the command Delete Modifications.

CREATING A FINAL FORMAT

1. Begin by choosing a file to format. Select a well-structured file with more than one branch that is more than one page long. Make this your current file and update it; after you have inserted the directives and printed your final copy, you may then choose to use the Delete Modifications command to delete the changes you made since the last update. If the file you decide to use already contains directives, you can either delete all of the old directives before creating the new format or carefully delete only those directives that will not be used for the new format.

2. We suggest that you print out a copy of the file to help you plan your modifications. Set viewspecs "mywGJ" to show all lines, all levels, statement numbers on the right, and spaces between statements.

Note: Use the Delete Text command to delete the following directives if you plan to use the same file containing the directives for the rough draft format.

```
.Ybs=2;
.Ybl=1;
.H="DRAFT";
.Fp=FR;
.Ifirst=5;
.Snf=On;
.Snftype=1;
```

The final format will be suitable for the printing of a master copy of a report or paper on a typewriter-quality terminal or a line printer. The format will be single spaced with no paragraph indenting and will have statement numbers on the right, a header, footer, and title page. We will list and define the directives for you to use to create your final format. Included are directives with which you are already familiar as well as new directives. The following is a list of the directives to use to create the overall format of the document:

Margins

.Lmbase=5;

Moves the absolute left margin 5 characters to the right. (Note that all other horizontal margins are calculated from Lmbase.)

```
    .Rm=63;
      Sets the right margin to 63 characters, counting from
      Lmbase.

  Vertical Spacing
    .Ybs=1;
      Sets the vertical distance between statements to equal
      one line.
    .Ybl=0;
      Sets the vertical distance between statements to equal
      zero blank lines.
    .Ypf=4;
      Sets the vertical distance between the bottom margin
      and the footer to equal four blank lines.
    .Yfh=4;
      Sets the spaces between header and the top margin of
      the text to four.

  Indenting
    .Ilev=2;
      Sets the number of spaces for level indenting to two.

  Statement Numbering
    .Sn=Off;
      Left statement numbers will not be printed.

    .Snfshow=<4;
      Show right statement numbers for levels one through
      three (the levels less than four).
    .Snf=68;
      Right statement numbers will be printed justified to
      68 characters from Lmbase.

  Pagination
    .Plev=1;
      Paginate before every statement of level one.
    .Pn=0;
      Set the current page number to zero.
    .Numdash=0;
      Sets the number of dashes that are ordinarily printed
      at the bottom of the page (when you use the command
      Output Terminal) to zero.  When you use perforated
      paper or letterhead on a typewriter terminal, it is
      unnecessary to have a series of dashes printed at the
      bottom of every page.
```

Header

.H="Title of the Paper";
The text within the quotes will be printed at the top of every page as a header.

.Hp=FR;
The header will be printed flush right.

Footer

.F="Organization Name.Split;page .Gpn;";
The text within the quotes (formatted by the directives included with the text) will be printed at the bottom of the page as the footer.

Note that two new directives are included within the footer. The directive Split justifies the text on the left of the directive to the left margin and justifies the text on the right of the directive to the right margin. The directive Gpn causes the current page number to be printed where the directive is. Both of these directives have a one time effect; however, when they are included within a header or footer, they take effect every time the header or footer is printed.

.Fp=FL;
The footer will be printed flush left.

3. Use the command Insert Text to insert the directives listed above in your origin statement, typing a space between each directive.

```
.....................................................................
You type:
          it0+e<CR> .Ybs=1; .Ybl=0; .Pn=0; .Plev=1; .Ilev=2;
          .Numdash=0; .Lmbase=5; .Rm=63; .Sn=Off; .Snfshow=<4;
          .Snf=68; .H="Title of the Paper"; .Hp=FR;
          .F="Organization Name.Split;page .Gpn;"; .Fp=FL;
          .Ypf=4; .Yfh=4;
You see:
          BASE C: Insert C: Text (to follow) A: 0+e
          T/[A]:  .Ybs=1; .Ybl=0; .Pn=0; .Plev=1; .Ilev=2;
          .Numdash=0; .Lmbase=5; .Rm=63; .Sn=Off; .Snfshow=<4;
          .Snf=68; .H="Title of the Paper"; .Hp=FR;
          .F="Organization Name.Split;page .Gpn;"; .Fp=FL;
          .Ypf=4; .Yfh=4;
.....................................................................
```

4. Print statement 0 to check for typographical errors. Make any necessary corrections.

5. You may choose to emphasize your level 1 headings by using the directive .Center; to center them or by using the Force Statement command to capitalize them.

6. You will now create a title page for your document. The format of a title page is very different from the general format of your document. Therefore, you will need to turn off most of the origin statement directives when you start creating your title page. We recommend that you include the title page, or any specially formatted pages, at the end of your file to avoid respecifying directives more than once.

The following is a list of directives and their definitions that you will use to create a title page.

Headers

.Hsw=Off;
The header will not be printed.

Footers

.Fsw=Off;
The footer will not be printed.

Vertical Spacing

.Gybl;
The number of lines specified will be generated before the line with this directive.

.Gyel;
The number of lines specified will be generated after the line with this directive.

.Vsplit;
Vertical space is inserted at the end of the line segment with this directive so that the rest of the statement will be justified to the bottom margin.

.Ybs=0;
Zero blank lines will be printed between statements

.Gcr;
Generate one carriage return.

Horizontal Spacing

.Sp=C;
The entire statement with this directive is centered.

```
Indenting
    .Ilev=0;
      Levels will be indented 0 spaces.

Statement Numbering
    .Snf=Off;
      Right statement numbers will not be printed.

Ignore Text
    .Igls;
      The text in the line segment that includes this
      directive will not be printed--directives will be
      followed.
```

7. Begin by inserting a top level statement at the end of your file with the heading "Title Page" and the following group of directives to turn off the general document format. The heading is simply for your convenience; when you follow it with the directive .Igls;, it will not be printed. (In our example we have used the general address 1.t to specify the tail of statement 1 or the last top level statement in your file.)

```
.....................................................
You type:
          is1.t<CR><CR>Title Page .Igls; .Hsw=Off; .Ybs=0;
          .Snf=Off; .Ilev=0;
You see:
          BASE C: Insert C: Statement (to follow) A: 1.t
          L:
          T/[A]: Title Page .Igls; .Hsw=Off; .Ybs=0; .Snf=Off;
          .Ilev=0;
.....................................................
```

The title page will automatically be printed on a separate page because the directive .Plev=1; that you inserted in the origin statement is still in effect.

8. Insert the text for your title page along with directives to format the text in a substatement following the title page heading, as in the following sample. Please replace the appropriate title, author or authors, date, and organization of your document for the corresponding text in our sample.

```
.....................................................
You type:
          is1.t<CR>d<CR>.Fsw=Off;.Sp=C;.Gybl=12;THE NAME OF
          YOUR DOCUMENT.Gyel=6;.Gcr;Author, Author 2
          (optional) and Author 3 (optional).Vsplit;.Gcr;
          Month Day, Year.Gyel=6;.Gcr;Department Name.Gcr;
          Organization Name.Gcr;City, State Zip Code
You see:
          BASE C: Insert C: Statement (to follow) A: 1.t
          L: d
          T/[A]: .Fsw=Off;.Sp=C;.Gybl=12;THE NAME OF YOUR
          DOCUMENT.Gyel=6;.Gcr;Author, Author 2 (optional)
          and Author 3 (optional).Vsplit;.Gcr;Month Day,
          Year.Gyel=6;.Gcr;Department Name.Gcr;Organization
          Name.Gcr;City, State Zip Code
.....................................................
```

Note: You must include the directive .Fsw=Off; in the substatement of your title page branch rather than in the top level statement. When you force pagination using the directives .Pbs; or .Plev;, the directive .Hsw; will take effect as if the statement that began the new page was actually on the preceding page (where it would have been if you had not forced pagination). This is true of several other directives such as .Hsw; and .Pn=n;. Therefore, if you included .Fsw=Off; in the top level statement, the footer will not appear on the last page of text before the title page. Note also that you must always follow the directive .Vsplit; with the directive .Gcr; to force the end of the line. If you do not add the .Gcr;, the Output Processor will complete the line with text from the rest of the statement before inserting the vertical space.

9. Use the command Print Branch to check the title page for errors. If your file is short, you may want to use the Print File command to recheck your entire file. Make any necessary corrections.

10. Now you are ready to print out a copy of your document. Jump to the origin statement of your file, and make sure to set your viewspecs to show all lines and levels. Use the command Output Terminal to print the file following directives. (The example here is for a typewriter terminal; if you have access to a line printer, see the chart at statement 6 for the appropriate command.)

```
.....................................................
You type:
          jo<CR>w<CR>ot<CR>nnny
You see:
          BASE C: Jump (to) C: Origin A: V: w
          BASE C: Output (to) C: Terminal OK/C: OK:
          (Send Form Feeds?) Y/N: (Simulate?) Y/N:
          (Wait at page break?) Y/N:
          (Go?) Y/N:
          Processing Output
.....................................................
```

11. After you have printed your file, check to see that all the directives have taken effect. If any of the directives have printed as text, check them carefully for typographical errors; a very common error is a missing beginning period or ending semicolon.

12. Optional: After you have printed a copy of the final format, you can make use of the directive .Grab=n;. The directive .Grab; allows you to force pagination if a specific number of lines will not all fit on the current page (the Output Processor will not force pagination if all the lines do fit on the page). When you look over your first copy of the final format, you may see headings at the bottom of page or paragraphs that have only one line on a page. The directive .Grab; counts from the first line of the statement that includes the .Grab;.

For each individual case, count the minimum number of lines you want to be on the same page (this number should include the blank lines between statements) and specify that number as the argument. For instance, .Grab=2; will ensure that the first two lines of a statement will be printed on the same page. To ensure that a heading, the blank lines that follow it, and the first two lines of next statement will all be printed on the same page, specify "4" for the argument. Use this directive to ensure that an entire table or chart will be printed on one page. Count the total number of lines, including blank lines, in the table or chart--if the table is 19 lines long, specify "19" as the argument. Use the command Replace Character to insert a .Grab; at the beginning of the statements that need them. Then print out the file using the command Output to Terminal (or the appropriate command from statement 6).

13. The following is a hypothetical example, replacing the first character of a heading, statement 6a, that begins with the word "The". BE SURE TO REMEMBER TO TYPE IN THE FIRST CHARACTER OF THE STATEMENT AFTER YOU HAVE TYPED THE DIRECTIVE .Grab;.

```
.....................................................
You type:
          rc6a<CR>.Grab=4;T<CR>
You see:
          BASE C: Replace C: Character (at) 6a
          (by) T: .Grab=4;T
.....................................................
```

Note that you need only specify the statement number to replace the first character in a statement. Use the back slash or print statement command to check for typographical errors. Make any necessary corrections. Using .Grab; can be a little tricky, because after putting the directives in all the right places and then printing out the file again, the pages will be redistributed to accommodate these instructions, and you may now discover new statements that need them. If you are concerned about this kind of formatting, you may have to print your file two or three extra times and insert new directives before every page is satisfactory.

14. You may now either choose to incorporate the directives into your file by using the Update File command or remove them from the file by using the command Delete Modifications.

PRINTING COMMANDS

Device	Command to follow directives	Command to ignore directives
Portable Terminal	Output Terminal (no formfeeds)	Print File
High-Quality Printing Terminal	Output Terminal (formfeeds)	Print File
Remote Terminal	Output Remote Printer	Output Sequential & (Tenex) Sendprint
Line Printer Without Spooler	Output Remote Printer	Output Sequential & (Tenex) Sendprint
Line Printer With Spooler	Output Printer File	Output Quickprint File
ARC Printer	Output Printer	Output Quickprint

INTRODUCTION

This sample session shows you how to use the NLS Format subsystem to impose a standard layout on an NLS file for local printing or for output to a phototypesetter. Since the Format subsystem is not available automatically, this sample session will show you how to call it through the Programs subsystem.

The Format subsystem automatically inserts Output Processor directives for each format. This scenario will teach you to format and print on a terminal or line printer the file you created in the "Editing Sample Session II". However, you may use the Format subsystem to format any other appropriate file. Beyond the standard formats, you can use Output Processor directives to control exactly the appearance of printed files, but hand modification of format requires special training. For further information on the use of Output Processor directives, see the "Document Formatting Sample Session".

Follow the step by step instructions, which include an example of the commands that you will be using and general comments. Using this scenario as a model, an inexperienced user should be able to perform any of the operations described here and refer to Help and other documentation for related information about formatting.

INSTRUCTION

1. The format shown in this sample session centers all top level statements so that the headings of chapters or sections in your document will be automatically centered. The statements in the file you created in Editing Sample Session II are all at the highest level. After you have logged in and used the Jump Link command to reach the file you created in Editing Sample Session II, begin by following the first example. Insert a top-level title for the document and move the remaining statements to follow it, down one level.

.....

You type:

```
is<CR><CR>EDITING REPORT<CR>
mg2<CR>9<CR>1<CR>d<CR>
```

You see:

```
BASE C: Insert C: Statement (to follow) A: L:
T/[A]: EDITING REPORT

BASE C: Move C: Group (from) A: 2
(through) A:9
(to follow) A: 1
L: d
```

.....

The Format subsystem automatically inserts Output Processor directives, instructions that control the format of a document, into your file. Update your file before beginning to use Format. After you have printed a copy of your document, you may want to use the Delete Modifications command. This command will erase all of the directives inserted by the Format subsystem and any other changes made since your last update.

2. Access to the Format subsystem is through the Programs subsystem. Use the following commands to reach it.

.....

You type:

eplpformat<CR>gf<CR>

You see:

BASE C: Execute (command in) C: Programs

PROG C: Load C: Program T/[A]: format

Loading User Program

Don't Execute via RUN PROGRAM Command

Use GOTO SUBSYSTEM Command

Loading User Program

Subsystem FORMAT Now Available (Attached)

BASE C: Goto (subsystem) C: Format OK:

FORM C:

.....

3. The Insert Format command inserts Output Processor directives into your file to create the format you specify. After typing the command words "Insert Format", you type <CR> to indicate that you want to format the file you have loaded. Typing "y" for yes will show you a list of formats available. You choose a format by typing its number. For this example use Format Number 1. The system will then prompt you for information to put on the title page. Note that several of the formats specify a type face and point size (e.g., 8 pt News Gothic). For further information on these formats, see the section entitled "Using the Photocomposition Formats" at the end of this sample session.

.....

You type:

if<CR>y1<CR>Editing Report<CR>HQU<CR><CR>

You see:

FORM C: Insert C: Format (in file at)
A: (using Format #)

List formats?

0: title page only

```
        1:  simple printer format
        2:  journal format
        3:  8 pt News Gothic, level one titles
        4:  9 pt Times Roman, level one titles
        5:  10 pt News Gothic, level one titles
        6:  8 pt News Gothic, lev 1 titles, lev 2 subtitles,
        right stmnt nums
        7:  9 pt Times Roman, lev 1 titles, lev 2 subtitles,
        right stmnt nums
        8:  8 pt News Gothic, level 1 titles, 2 columns; you
        will have to hand format to balance columns at end of
        each branch
        9:  9 pt Times Roman, level 1 titles, 2 columns; you
        will have to hand format to balance columns at end of
        each branch
        10: 9 pt Times Roman, indented paragraphs, no statement
        numbers
        11: ARC userguides format

        .........Format # T/[A]: 1

        (Title:) T/[A]: Editing Report

        (Author Ident(s):) T/[A]: HQU

        (Journal Number:) T/[A]: <CR>

        (Formatting File)
```

.....

How to respond to the prompts:

A: Although you are using your loaded file in this step, you can format a file you do not have loaded. Type in the name of your chosen file at the first prompt A:, and the file you have specified will be formatted.

List: You need not see the list of formats every time you use the command. If you type "n" for "no" instead of "y" for yes at that step, you will simply be prompted for a format number and the list of formats will not be printed. For more information about the formats, use the Help command.

Author: If the author has an IDENT (we used the hypothetical IDENT HQU) the system will gather his or her name and address and add them properly to the title page. If the author is unknown to the Ident system, you may type

in his or her name and address. To use carriage returns to end lines in the name and address, you must precede them with <CTRL-V> to prevent them from interrupting your command.

Number: The Augmentation Research Center maintains a numbered and automatically cataloged online collection of documents called the Journal. The simple format chosen in this sample session does not require a journal number. For the process of getting a reserve journal number, ask for "preassigned number" with the Help command. If you do not have a journal number, you may hit <CR> and continue. The place usually occupied by the journal number in the layout will then be blank.

4. You must leave the Format subsystem to see your formatted file.

```
.....................................................................
You type:
          qtb<CR>
You see:
          FORM C: Quit OK/C: To C: Base OK:
          BASE C:
.....................................................................
```

5. To see most of the directives that were inserted by the Format subsystem, print statement 0.

```
.....................................................................
You type:
          ps0<CR><CR>

You see:

   BASE C: Print C: Statement (at) 0 V:
   <DIRECTORYNAME,EDITING.NLS;#> DATE TIME IDENT
   ;;;;LM=-3;.SN=0;.RM=72; .BRM=68; .SNF=72; .SNFShow=<=3;
   .YBS=1,6p; .YBL=0,2p; .F="page .GPN;";.H1="Editing Report";
   .PN=0; .PES; .FP=FR;. .PxPShow=1; .PxFShow=1,2; .PxFYD=1;
   .PxFYS=2; .PxFYU=2;
.....................................................................
```

The expressions bounded by a period on the left and a semicolon on the right are directives. For example, RM=72 sets the right margin to 72 characters. To learn more about directives, see the Output Processor Users' Guide or type "directives" in the Help command.

6. You are now ready to print your file at your terminal, on a line printer, or on a high-quality printing terminal. Depending on what kind of terminals and printing equipment are available to you, there are various commands that you can use to print a formatted version of your file. Output Processor directives are followed as instructions and do not appear as text when you print your file using the commands Output Terminal or Output Printer. They are ignored as instructions and printed as text when you print a file with the commands Print File or Output Quickprint. The following example uses the command Output Terminal for printing at your terminal. For further information on printing files with or without directives, see the Print Commands table at the end of this section.

```
.....................................................
You type:
          ot<CR>nnny
You see:
          BASE C: Output (to) C: Terminal OK/C: OK:
          (Send Form Feeds?) Y/N:  (Simulate?) Y/N:
          (Wait at page break?) Y/N:
          (Go?) Y/N:
          Processing Output
.....................................................
```

A formatted version of your file will print out, followed by a title page that was created by the Format subsystem. Note that when you use the command Output Terminal, page breaks are indicated by a series of dashes.

7. All the changes you made in the file since the last time you used the Update command can be removed with the Delete Modifications command. If you updated just before inserting format, you can remove the directives and the title page by using the command Delete Modifications or you may choose to permanently incorporate the directives into your file by updating again.

```
.....................................................
You type:
          dm<CR><CR>
You see:
          BASE C: Delete C: Modifications (to file) OK:
          (really?) OK:
          BASE C:
.....................................................
```

PRINTING COMMANDS

Device	Command to follow directives	Command to ignore directives
Portable Terminal	Output Terminal (no formfeeds)	Print File
High-Quality Printing Terminal	Output Terminal (formfeeds)	Print File
Remote Terminal	Output Remote Printer	Output Sequential & (Tenex) Sendprint
Line Printer Without Spooler	Output Remote Printer	Output Sequential & (Tenex) Sendprint
Line Printer With Spooler	Output Printer File	Output Quickprint File
ARC Printer	Output Printer	Output Quickprint

USING THE PHOTOCOMPOSITION FORMATS

Photocomposition (sometimes referred to as COM) is the method of typesetting that you can use to create high quality, camera-ready copy of an NLS file for printing. Besides the professional appearance of the document and the high-quality type for good reproduction, there are several advantages to using phototypesetting rather than a high-quality typewriter terminal or line printer. Some of these are the ability to specify one or more of several proportionally spaced serif or sans serif type faces and sizes, columnation, full justification, and the ability to fit as much as 40 percent more text on a page.

Most of the formats in the Format subsystem were designed particularly to be output to a phototypesetter. The Output to COM command creates a sequential file that can be read by a phototypesetting machine. At the present time, you can choose from a list of formats that include Times Roman or News Gothic type, full justification, or columnation. Several vendors provide us with phototypesetting. We suggest that you consult your client coordinator at ARC about choosing one, since the different phototypesetting machines result in wide variations in the type.

1. Begin by entering the Format subsystem. Choose a format that is appropriate for your document. The Output Processor Users' Guide has samples of the various type faces. Just as you did in the previous section, specify the corresponding number of your chosen format when you are prompted.

2. After inserting the format, leave the Format subsystem. You are now ready to create the file that will be sent to the phototypesetter. At this time, you will have to specify the vendor you have chosen. An example for the command to use for each vendor follows:

Command for George Lithograph, San Francisco, California

```
.....................................................
You type:
          ocs<CR><CR>
You see:
          BASE C: Output (to) C: Com (for device)
          OK/C: Singer OK/C: OK:
          Processing Output
.....................................................
```

Command for DDSI, Los Angeles, California

```
.....................................................
You type:
          occ<CR><CR>
You see:
          BASE C: Output (to) C: Com (for device) OK/C: Comp80
          OK/C: OK:
          Processing Output
.....................................................
```

Command for Data Composition, San Francisco, California

```
.....................................................
You type:
          ocv<CR><CR>
You see:
          BASE C: Output (to) C: Com (for device) OK/C:
Videocomp
          OK/C: OK:
          Processing Output
.....................................................
```

3. The file you have created will not be in your directory; the command Output Com automatically places it in the COM directory. You can make sure your file is in the COM directory by using the Show Directory command. When you are ready to send your file to the phototypesetter, send a message to Feedback. Be sure to specify the name of the file, the vendor you have chosen, whether you want proofs or camera-ready copy, and your name and address. The ARC staff will record your file on tape, the tape will be delivered to the vendor, and the type will be sent to you by mail. The billing procedure for this service should be worked out with your client coordinator.

SAMPLE SESSION SUMMARY

To enter the Format subsystem

Goto Programs, Load Format, Goto Format.

Commands in Format:

Insert Format adds directives and a title page so that a file can be printed in one of a list of formats. This list changes as ARC adds new formats.

Another command, Delete Directives, will remove all the directives from a file. You could have used it instead of Delete Modifications to create a file without directives for easy online reading; or you can use it to remove old directives from a file and start fresh with a new format. Note that Delete Directives will not remove the content of the new title page branch created by Insert Format.

After you have formatted the file:

To see a formatted printout use the Output Terminal command or the appropriate command from the Printing Comands chart at the end of this sample session.

To remove the format:

As an alternative to the Delete Directives command, you may use the Base command Delete Modifications, which removes all changes made since the last time you used the Base Update command.

For assistance when using Format:

Use <CTRL-Q> or the Help command. (See the Help Services Sample Session.)

INTRODUCTION

The online computer system you will be using provides direct response to what you have just typed into your terminal. It also provides help services, various types of feedback which you can request for immediate, online information about all aspects of the system. Three help services include the following:

1. Lists of command alternatives you may use at any point (reached by typing a question mark).

2. Explanations of these alternatives (reached by typing a <CTRL-Q>).

3. Descriptive information, such as definitions, explanations, and instructions (reached by giving the Help command).

INSTRUCTION

1. You have logged in and are ready to use NLS. Let's first use the Help command to see how it describes itself. To execute the Help command you type "h" for Help, and then the term you want to have explained. In this case, you want to learn more about the command "Help."

.....
You type:
hhelp<CR>
You see:
BASE C: Help OK/T: T: help
.....

This command will produce an explanation of Help at your terminal, followed by a list of items called a "menu" which you can use to obtain further information. Below the menu you will see the following line:

(HELP)</T: T:

As explained in your Help typeout, this line prompts you to ask Help for more information.

2. Let's use the menu to learn more about help services. Menu item 3 looks like it will supply us with this information.

.....
You type:
3<CR>
You see:
Help </T: T: 3
.....

A description of how to use <CTRL-Q> and <CTRL-S> appears, followed by a reference to using the question mark for help. Type in "question mark" to learn more about it.

.....
You type:
question mark<OK>
You see:
Help </T: T: question mark
.....

3. Now let's try a practical exercise with the Help command to aid us in working online. After having read the "Preface to NLS

Tools" you know that you need a workspace called a file. You can use Help to discover how to set up your own files.

```
.....................................................
You type:
          files<CR>
You see:
          (Help) </T: T: files
.....................................................
```

This is followed by the most general explanation of files in Help. For more specific details refer to the menu items, or type in a term discussed in the explanation.

4. Since you want to know how to go about creating a file, and none of the menu items directly refers to this kind of information, you need to look for a new term to type in. The description mentions a command that creates a file. You might choose to type in this command name.

```
.....................................................
You type:
           create<SP>file<CR>
You see:
           (Help) </T: T: create file
.....................................................
```

This produces the syntax of the command, followed by an explanation of how the command works, references, and a menu. If you type a 1<CR>, you can see an example of how the command looks when implemented at your terminal. Note: You could have left off the word "file" in this case, but sometimes more than one word is necessary.

5. You now have enough information to create a file. To leave Help, type a <CTRL-X>.

```
.....................................................
You type:
           <CTRL-X>
You see:
           (Help) </T:

           BASE C:
.....................................................
```

Typing <CTRL-X> takes you back to Base, as indicated by the prompt "BASE C:".

6. You create a file by use of the information you have learned, and name the file FIRST-AID.

```
.....................................................
You type:
          <SP>crffirst-aid<CR>
You see:
          BASE C: Create  C: File T: first-aid

          <DIRECTORYNAME, FIRST-AID.NLS;1,>
.....................................................
```

7. Now you have a workspace, but what can you do next? You can see a list of your current possible alternatives if you type a question mark.

```
.....................................................
You type:
          ?
You see:
          BASE C:
          Current Alternatives are:
.....................................................
```

Below the heading appears a long list of command words (the "verb" type) that you may use at this time.

8. Scanning the command words, Insert looks like the most promising choice. Notice that a broken line has ended the list. At this point you can type the first letter or a space and the first letter of one of the commands and it will become part of your command. If you type <CR>, you return to where you were before you typed a question mark.

```
.....................................................
You type:
          i
You see:
          ---Insert C:
.....................................................
```

You now have another prompt (C:) telling you that another command word is necessary. Again, you can type a question mark to see the commands available to you.

```
.....................................................
You type:
          ?
You see:
          ---Insert C:
          Current Alternatives are:
.....................................................
```

This heading is followed by a list of command words, this time the "noun" type.

9. You choose the word Statement since you know that a statement is the basic component of NLS files.

```
.....................................................
You type:
          s
You see:
          ---Statement (to Follow) A:
.....................................................
```

You are again prompted to continue your command, this time with an A:, which you recognize is the prompt for ADDRESS. You type in a 0, since you want your statement to follow the header (name of the file) which is always statement 0.

```
.....................................................
You type:
          0<CR>
You see:
          ---Statement (to follow) A: 0
          L:
.....................................................
```

10. After you type in your address, another prompt appears (L:) which does not look familiar to you. Having had so much luck with using the ?, you try it once again.

```
.....................................................
You type:
          ?
You see:
          L:
          Please type a LEVEL-ADJUST String:
.....................................................
```

11. Alternatives follow the line requesting a LEVEL-ADJUST, but this time you probably do not know enough about the alternatives

for them to be helpful to you. There is a help service that provides a way to get detailed information about the command you are using at the moment. You can obtain this information by typing a <CTRL-Q>.

```
.....................................................................
You type:
          <CTRL-Q>
You see:
          BASE
          ()
          LEVEL-ADJUST:
.....................................................................
```

This is followed by a Help description which both explains the term and tells you what to do when presented with the prompt L:. You can now complete your command.

12. When you use <CTRL-Q> you are taken into a Help description and placed in the Help command. To return to Base from the Help command, you must now type the same <CTRL-X> that you used in step 5.

```
.....................................................................
You type:
          <CTRL-X>
You see:
          (Help) </T:
          BASE C:
.....................................................................
```

13. Typing <CTRL-Q> aborts any command you were typing in when you hit <CTRL-Q>. Therefore you are now back at BASE C:, ready to insert a statement.

```
.....................................................................
You type:
          is0<CR>You can get Help by typing an h at
          the herald or at BASE C:, followed by a
          term.<CR>
You see:
          BASE C: Insert C: Statement (to follow) A: 0
          L:
          T: You can get Help by typing an h at the
          herald or at BASE C:, followed by a term.
.....................................................................
```

14. After considering your first statement, you decide to add more to it. You know that Insert is the correct first term in your command, but you do not want to begin a new statement. To check other alternatives:

```
.....................................................
You type:
          i?
You see:
          BASE C: Insert C:
          Current Alternatives are:
.....................................................
```

15. Scanning the alternatives, Text looks like the most likely choice, but you are not sure what its effect will be. Two ways to obtain this information are available. The first, illustrated here, is to type a "t" for Text, then hit the <CTRL-Q>. An explanation of the command will follow.

```
.....................................................
You type:
          t<CTRL-Q>
You see:
          ---Text (to follow) A:
          BASE
          ()
          TEXT: Insert Text (to follow) DESTINATION CONTENT
          OK:
.....................................................
```

The second way to obtain this information is to type a <CTRL-X>, then use the Help command, typing in the words "insert text". Try this if you want more practice.

16. You return to Base and implement your command. You add text in the form of a sentence to statement 1 that will tell more about using help services. Since you want your new text to come at the end of the statement, your ADDRESS will be the statement number and "+e". (See the "Editing Sample Session I" for more information on adding text.)

.....

You type:

<CTRL-X>it1+e<SP>Using <CTRL-Q> also takes you to Help, whereas typing a ? shows you a list of Current Alternatives.

You see:

BASE C: Insert C: Text (to follow) A: 1+e
T: Using <CTRL-Q> also takes you to Help, whereas typing a ? shows you a list of Current Alternatives.

.....

To see how the completed statement now looks, type a \.

.....

You type:

\

You see:

BASE C: \
1 You can get Help by typing an h at the herald or at Base C:, followed by a term. Using CTRL-Q also takes you to Help, whereas typing a ? shows you a list of Current Alternatives.

.....

17. You have completed the sample session and are familiar with help services. You can now delete your First-Aid file.

.....

You type:

dffirst-aid<CR><CR>

You see:

BASE C: Delete C: File T: first-aid
OK:

Deleted Files Are:
<DIRECTORYNAME,FIRST-AIDE.NLS;1,> and its partial copy

.....

GLOSSARY OF NLS TERMS

A:
This prompt indicates that you are typing in the Address of a specific place in a file. Statement numbers and SIDs are types of Addresses that may be used.

ADDRESS
The location of a specific place in a file. You may type either a statement number or an SID to indicate a particular statement. End the Address with an OK.

APPEND STATEMENT
Use the command "Append Statement" to attach the statement you bug to another statement. The first statement specified is added to the end of the second statement specified. The characters you specify for "join with" will be inserted between the text of the two statements. For example, if you append two paragraphs, join them with two spaces for the double spaces between sentences. If you do not want to add characters between the two appended statements, type <CTRL-N> after "join with". <CTRL-N> is a NULL character; it means "nothing." Substructure will follow an appended statement. Type "as" for Append Statement.

B:
This prompt indicates that you can BUG an entity on the screen by pressing the right mouse button or the OK key when the cursor is positioned correctly.

B/A:
This prompt indicates that you can BUG an entity on the screen by pressing the right mouse button or the OK key when the cursor is positioned correctly, or type in the Address of a specific place in a file. The slash means "or".

B/C:
This prompt indicates that you can BUG an entity on the screen by pressing the right mouse button or the OK key when the cursor is positioned correctly, or type a command word.

B/T:
This prompt indicates that you can BUG an entity on the screen, by pressing the right mouse button or the OK key when the cursor is positioned correctly, or type some text in from the keyboard.

BACK
The statement immediately preceding the statement that you BUG, regardless of level. This command word is a noun that must be preceded by the verb "Jump." Type "jb" for Jump (to) Back.

BACKSPACE CHARACTER
One character of input is deleted each time the BACKSPACE key on the keyboard or the left mouse button is pressed. A BUG or command word can also be deleted with a BACKSPACE key.

BACKSPACE WORD
The last word typed in or command word is deleted each time the BACKSPACE WORD key on the keyboard is pressed.

BRANCH
A logical section of a hierarchically structured file that includes a statement with all its substatements, all their substatements, and so on to the end. This command word is a noun that must be preceded by a verb such as Move, Delete, or Replace. Type "b" for Branch.

BREAK STATEMENT
Use the command "Break Statement" to divide one statement into two statements. The statement will break at the first non-printing character following the printing character you BUG. The text preceding the "break" will remain as the original statement; the text following the break will be a new statement that follows the original statement at the level you specify. Any substructure will remain with the original statement. Type "bs" for Break Statement.

BUG
To "bug," use the mouse to control the cursor or traveling mark on the screen. Position the cursor under a particular character on the screen and press the right mouse button. The letter will be highlighted or covered with a bright rectangle to indicate which character you have selected.
When specifying text, the first and last characters must be BUGGED.
When specifying a Group, the first and last branches must be BUGGED.

C:
A prompt that indicates you can type a command word.

CHARACTER
Characters are single elements such as a letter, number, punctuation mark, carriage return, or space. As a command word, this is a noun that must be preceded by a verb as in the Delete Character command. Type "c" for Character.

COMMAND ACCEPT or OK
Pressing the right mouse button or the OK key on the keyboard indicates you are finished specifying a command or part of a command (such as viewspecs or text you type in). OK: is a prompt indicating that you are to do this.

COMMAND DELETE
Pressing the CMD DEL key on the keyboard or the middle mouse button aborts any command before you finish specifying it. After typing either, you may specify a new command. <CD> is also written to denote COMMAND DELETE.

COMMAND RECOGNITION
The system recognizes a command word when you have typed enough to make the word unique. When it recognizes the command word, NLS fills out the rest of the word and then prints helpful words in parentheses to let you know what is expected next. It will also prompt you for what you should type in next.
For frequently used command words, type only the first character and the rest will be recognized and completed for you.
When two or more commands begin with the same letter, you often need to type a space and then enough of the command so that the computer can recognize and complete it. For example, there are several command words that begin with "c": Copy, Create, and Connect. Type "c" for Copy, "<SP>cr" for Create, and "<SP>co" for Connect. <SP> is written to instruct you when to type the space bar.

CONTROL CHARACTER
A non-printing character that has a special function, such as as <CTRL-C>. Type a control character by striking the character after the dash (in this example "c") while holding down the CTRL key (like using the SHIFT key for uppercase letters.) The brackets indicate a non-printing character.

COPY

Use the "Copy" command to reproduce the Character, Word, Text, Statement, Branch, or Group that you BUG or address. This command word is a verb that must be followed by the noun that tells what you want to copy. Type "c" for Copy.

CREATE

This command word is a verb that must be followed by the noun "File." Type "<SP>crf" for "Create File." The command Create File makes a new file in your directory; the file will have the name you type in after the B/T: prompt, and an origin statement will be automatically created.

CURSOR

This is the traveling mark on the screen that is controlled by the mouse. See BUG.

DELETE

Use the "Delete" command to remove a Character, Word, Text or Statement from your file. Delete is a verb that must be followed by a noun that specifies the entity to be removed. Type "d" for Delete. Blank spaces will be taken out as necessary after a deletion and the screen will be re-created to reflect the changes.
Other entities you can delete are Branch and Group.

DIRECTORY

A directory is the user's work space in the computer. It may belong to one person or be shared by a group. When a user logs in to the computer, the directory name identifies her and allows her to open up files in her work space. The directory list is a library of the user's files. Use the Show Directory command to see the list of files in your workspace.

DISPLAY

This piece of equipment is the television-like screen on which the user can see part of the contents of a file. There is an on/off button located on the back or side of the display. The contrast between the characters and the background can be adjusted. The display is connected to the computer through the line processor.

DOWN

The statement one level below an indicated statement. This command word is a noun that must be preceded by the verb Jump. Type "jd" for Jump (to) Down.

END OF BRANCH

The very last statement in the branch specified. It's level doesn't matter. Precede the command word End with the command word Jump. Then BUG the branch whose "end" statement you want to display at the top of the screen. The very last statement in the file will be displayed if you BUG the origin statement. Type "je" for Jump (to) End.

ESCAPE OR <ESC>

When you press this button on the keyboard after typing enough characters of a file name to uniquely differentiate it from the other files in your directory, the system will automatically fill in the rest of the file name in link format. If there is more than one version of the file, the system will fill in the highest existing version. When you have not typed enough characters to make the name unique or if the file does not exist, you will see "<ESC>".

FILE

A user may fill her workspace with files that store information. The file may be a mailbox, a collection of personal notes, an article, a letter, a program, or a data base.

Use the Create File command to make new files. The Jump Link command will access a file that you have previously created. See DIRECTORY

FILENAME

A file is named by the user upon creation. The name can be any short word or phrase that may include dashes for readability. However, it cannot contain spaces, commas, periods, or semicolons. The filename is later used to access the file for reading or writing.

GENERATION

The number of the successive versions of each file. This number appears in the complete filename: "NAME.NLS.3". The Create File command makes generation 1 and each successive Update File command, makes the next higher generation out of your last generation and modifications to it. When you Jump (to) Link and type the name of the file, the highest generation will be displayed. See also ESCAPE and SHOW DIRECTORY.

GROUP

A series of consecutive branches at the same level with the same source. You may indicate the first branch and the last branch in the group with BUGS or addresses. This command word is a noun that must be preceded by a verb such as Move, Delete, or Replace. Type "g" for Group.

HERALD

The signal the computer types when it is ready for the user to give a command. @ is the executive herald. BASE is an NLS herald.

INITIAL FILE

This is a file named with your ident that is automatically loaded when you type "DNLS <CR>". Keep this file as it is necessary for your use of NLS. The kinds of information people often have in their initial files include "to do" lists and reminders, links to other files, and process commands branches, as well as one's mailbox.

INSERT

Use the "Insert" command to create, input, or duplicate new information in a file. Insert is a verb that must be followed by a noun that specifies the entity to be added, such as Statement or Word. Type "i" for Insert.

JUMP

Use the "Jump" command to move from one point in a file to another, or from one file to another. Jump is a verb that must be followed by a noun or a BUG. Type "j" for Jump.

JUMP (TO) LINK

The Jump (to) Link command is used to move around within a file by specifying the location of the statement you want to see. One kind of link is a statement number. The Jump (to) Link command is also used to move to a different file than the one on the screen. Type in the name of the file you wish to see, and be sure to type in a comma after it. The comma tells the computer to look for a file by that name. Type "jl" for Jump (to) Link.
See also ESCAPE for informations on a shortcut in typing filenames.

JUMP (TO) RETURN

Use this command to go back to the last view you had in your file window. Type "jr" for Jump (to) Return. The system will prompt you with the last screenful and "Y/N:". Type "y" for yes if you want to return to that view. If you type "n," the system will prompt you with the first few words of your next-to-the-last screenful and a "Y/N:". This "return ring" will go into a complete circle, including your current view. After you have selected a return view, type OK, and your screen will be recreated with the new file displayed.

KEYBOARD

The keyboard is one of the input devices that is connected to the computer through the line processor. Although DNLS can support many kinds of keyboards, you will probably be using a Data Media, which has been found to be reliable and easy to use. The numbers and letters on the keyboard are arranged like a standard typewriter keyboard, with special black keys on either side. The black keys above the keyboard probably should not need to be changed once your keyboard and display are adjusted properly.

KEYSET

The keyset is a device with five piano-like keys for typing characters to NLS at a display work station. The keyset is an alternative to the keyboard and is designed to facilitate rapid editing. With your left hand on the keyset and your right on the mouse, you can give input to the system without ever moving your hands back to the keyboard. This allows you to keep your eyes on the screen while quickly specifying comands, moving around in files, and changing views. Using the keyset is optional. The keyset is connected to the computer via the line processor.

L:

L: is a prompt that indicates you can adjust the level (either up or down) of a statement. The level adjust is relative to the level of the statement that you reference. Type "d" for down or "u" for up, or "uu" to go up two levels, etc., and end with an OK. If you do not want to change the level or if you want the statement to be at the same level as the referenced (BUGGED) statement, just type an OK.

LEVEL

A statement's position in the hierarchical structure of a file. The origin statement is the only statement at the highest level (level 0). Statements 1, 2, 3, etc. are subordinate to the origin, and are said to be at the "top level" or "level 1". Their substatements (1a, 1b, 2a, 2b etc.) are at the second level. Levels down to 64 are possible. The level of a statement's can be adjusted up or down relative to the statement it follows.

LEVEL-ADJUST

When doing the Insert Statement command, specify the level of the new statement relative to the one you BUGGED for it to follow. After the L:, type OK for the same level, type "d" for down a level (substatement), type "u" for up a level or type "uu" for up two levels, and so forth. See L:

LINE PROCESSOR

The line processor is a microcomputer which processes data to and from the computer. The four silver toggles on the front of the line processor should all be down. If any of the four numbered lights on the top start to flash, press the System Reset button. All the work station devices are plugged into the line processor.

LINK

A location in some file. For instance in response to the T: in the Jump (to) Link command, you can type in the name of a file (followed by a comma) or a statement number. See Jump (to) Link.

LOGIN

The instructions you give to the computer to identify yourself and gain access to your files is called "logging in".

LOGOUT

Use the "Logout" command to leave both the NLS system and the TOPS-20 Executive Operating system at the same time. Type "<SP>1" for Logout when in NLS.

MOUSE

The mouse is a hand-sized device with three buttons on the top. It rolls freely on any flat surface, moving the cursor on the display screen correspondingly. For example, when you move the mouse to the right, the cursor mark on the screen moves to the right also. The three buttons can be used alone or with the keyset or keyboard for different functions. The mouse is plugged into the line processor.

MOVE

Use the "Move" command to rearrange information in your file. For example, you can move a single character to follow another or move an entire statement from the beginning of your file to the end of your file. Move is a verb that must be followed by a noun that specifies the entity to be moved. Type "m" for Move.

NEXT

The statement immediately following the statement that you BUG, regardless of level. This command word is a noun that must be preceded by the verb "Jump". Type "j<SP>n" for Jump (to) Next.

NOISE WORDS

After typing a command word, the system may respond with a word or expression in parentheses to help you understand the purpose of the command. These "noise words" will be followed by a prompt to indicate what you do next.

OK:

This prompt indicates that you are to press the right mouse button or the OK key on the keyboard. Use OK to end fields in a command.

OK/C:

This prompt indicates that you are to type OK or to give another command word.

OK/T:

This prompt indicates that you are to type OK or to type the appropriate information on the keyboard.

ORIGIN STATEMENT

The first statement in a file.
Every file begins with a statement created automatically by the system. It is the source of all the statements in the file, one level above the first level statements. The origin statement contains the name of your file and the date, time, and ident of the last person to update the file.

PREDECESSOR

The statement of the same level that precedes the statement that you BUG. The predecessor and the statement must share the same source statement; not every statement has a predecessor. If there is no predecessor, the statement you BUG will move to the top of your screen. This command word is a noun and must be preceded by the verb "Jump." Type "jp" for Jump (to) Predecessor.

PROMPT

Prompts are the upper case letters followed by colons that appear in the command window. These symbols are printed by the system to indicate what the user can type next. For example, the prompt "C:" indicates that the user is expected to type in a command word. If you have a choice, the prompts for each alternative are separated by slashes indicating "OR."

REPLACE

Use the "Replace" command to change information in a file by putting new information in its place. Think of this as a combination of the functions Delete and Insert. Replace is a verb that must be followed by a noun that specifies the entity to be removed. Type "r" for Replace. If your replacement contains more or less characters or statements than the original information, the system will automatically adjust the surrounding text or statements.

RETURN RING

The system automatically maintains a list of the last ten "screen-fulls" you had in your file window. When you use the command "Jump to Return," the full circle of views, from the tenth-to-the-last view on to the current view, is referred to as the "return ring."

SET VIEWSPECS

Use this command to change your viewspecs for the current NLS work session. Type "<SP>sev" for Set Viewspecs. See VIEWSPECS.

SHOW DIRECTORY

Use this command to display a list of the files in a directory. Type <SP>shd for Show Directory. Following the OK/T: prompt, simply type an OK to see the files of your own directory. End the command with another OK.

SID (STATEMENT IDENTFIER)

The system automatically assigns a unique number to every statement as it is inserted into a file. These numbers always begin with a zero and they reflect the order in which the statements were created (e.g., 03, 0214). Every statement always retains the same SID, regardless of editing changes. If the statement is deleted, the SID number is not used again for another statement. To see SIDs, use viewspecs "mI." You may use an SID to indicate the location of a statement, instead of BUGGING it.

SOURCE STATEMENT

Every statement (except the origin statement) belongs to a statement that is one level higher, called its source statement. For example, a third level statement has a second level source statement, and a first level statement has the origin statement as its source.

STATEMENT

The statement is the basic structural unit of an NLS file. It may be a single character, a word, a title, a heading, some text, or a paragraph. All characters in an NLS file are in some statement. It is an entity that can be treated as a separate unit. You can point to a statement by bugging any character in the statement. Statement is a command word that must be preceded by a verb; type "s" for Statement.

STATEMENT NUMBER

A statement number is associated with every statement in a file. This number reflects the statement's position in the hierarchical structure of the file. You can display these numbers by setting viewspec m. A statement number is not permanent, and if you change the position of the statement, the statement will automatically be renumbered. A statement number contains alternating numbers and letters, and always begins with a number.

You may use a statement number to indicate the location of a statement you want a command to act upon, instead of BUGGING it.

STRUCTURE

The hierarchical outline format of a file is its structure.

SUBSTATEMENT

A substatement is any of the statements that belongs to and is one level below a statement. For example, the statement numbered 1a is a substatement of statement 1. Not all statements have substatements.

SUBSTRUCTURE

All of the statements subordinate to a statement are considered its substructure; in other words, all the statement's substatements, and all their substatements in turn, and so on.

SUBSTITUTE

Use the command Substitute to replace a Character, Word, or Text EVERY where it appears in a Statement, Branch, or Group. This is often referred to as a "global" substitution. The command word Substitute is a verb that must be followed by a noun and an object, such as Substitute Word in Group or Substitute Text in Branch. Type "s" for Substitute. The following is an example for Substitute Text in Statement. First, type "sts" for Substitute Text in Statement, and then bug the appropriate statement. You will now be prompted to type in the "new text" that you want to replace and then the "old text" that you want to insert in its place. After ending the text with OK, you will see a S/Y/N: prompt. Type "y" for yes if you are finished making substitutions in this particular statement; if you type "n" you will be prompted again to type in new text and old text and so on until you are finished. Type "s" for feedback on what you have specified.

SUCCESSOR

The statement at the same level that follows a statement that you BUG. The successor and the statement must share the same source statement, and not every statement has a successor. If a statement does not have a successor, then the statement you BUG is moved to the top of the screen. This command word is a noun that must be preceded by the verb "Jump." Type "js" for Jump (to) Successor.

T:

Indicates that you can type the appropriate information from the keyboard.

TEXT

Text is a string of one or more contiguous (adjacent) characters within a single statement that you specify by bugging the first and last character. Text is a command word that must be preceded by a verb; type "t" for Text.

TRANSPOSE

Use the command Transpose to make the two text entities that you bug switch places. This command word is a verb that must be followed by a noun, such as Character, Word, Text, Statement, Group, or Branch. Type "t" for Transpose.

UP

The source statement one level higher than the statement you bug or address. This command word is a noun that must be preceded by the verb "Jump." Type "ju" for Jump (to) Up.

UPDATE File

Use the "Update File" command to incorporate into your file the modifications you have made since its creation or last update, making a permanent copy with all changes or additions. Type "uf" for Update File.

V:

This prompt indicates that you can change the view of the information on your screen. To keep the same view, type OK. Grab=2;To change the view at the V:, type one or more viewspec letters, ending with OK.

VIEWSPECS

Viewspecs are single-letter codes that control the appearance or "view" of your files. Some viewspecs are set automatically when you log in. These are called the default viewspecs. When viewspecs are allowed in a command, you are prompted with V:. You may type in a string of any viewspec codes, terminated by OK. Type just an OK if you don't want to change the viewspecs. Uppercase viewspecs do different things than lowercase viewspecs.

a: show one less level.

b: show one more level.

c: show all the statements in the file.

d: show only the first level of the hierarchical structure of the file.

m: show numbers with statements.

n: do not show numbers with statements.

s: show all lines of the levels displayed.

t: show the first lines only of all levels displayed.

y: show the blank lines between statements.

z: do not show the blank lines between statements.

G: number statements on the right.

H: number statements on the left.

I: show statement numbers, not SIDs.

J: show SIDs, not statement numbers.

VERSION

The number of the successive generation of each file. This number appears in the complete filename: "NAME.NLS.3". The Create File command makes version 1, and each successive Update FIle command makes the next higher version out of your last version, and modifications to it. When you Jump to Link and type the name of the file, the highest version will be displayed. See also ESCAPE and SHOW DIRECTIORY.

WINDOWS

The display screen is divided into several areas, called windows. Each window contains specific useful information as described below:

COMMAND WINDOW: The command being specified is displayed here with noise words and prompts as you type it.

FILE WINDOW: The file window displays files or parts of files.

STATUS WINDOW: This window provides feedback and shows interaction with the TOPS-20 Executive Operating system. Once information is displayed in the status window, it remains there until a screen re-creation or new message appears. It is used for error messages, system messages, and the name of the file being jumped to or updated.

TYPEIN WINDOW: This window displays the typein or address portion of a command as it will be accepted after the OK.

VIEWSPEC WINDOW: This window contains characters that indicate the current status of certain viewspecs. This section of the screen will brighten when the viewspecs can be changed.

WORD

A word is a contiguous string of letters and/or numbers, bounded by spaces and/or punctuation marks. NLS understands that punctuation marks are not part of a word. Word is a command word that must be preceded by a verb; type "w" for Word.

JUMP TO BACK

Use this command to place the statement which immediately precedes the statement you BUG, regardless of structure, at the top of the file window on your screen. You might usually BUG the statement currently at the top of the screen in order to see what is before it. You will view a different portion of your file: the "back" statement plus as much of what follows as can fit.

At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: Back B/A: V:

JUMP TO SUCCESSOR

Use this command to place the statement which FOLLOWS the statement you BUG, at the same level with the same source, at the top of the file window on your screen. You will view a different portion of your file: the successor plus as much of what follows as can fit.

At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: Successor B/A: V:

JUMP TO PREDECESSOR

Use this command to place the statement which PRECEDES the statement you BUG, at the same level with the same source, at the top of the file window on your screen. You will view a different portion of your file: the predecessor plus as much of what follows as can fit.

At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: Predecessor B/A: V:

JUMP TO UP

Use this command to place the SOURCE of the statement you BUG (i.e., the statement one level up from it) at the top of the file window on your screen. This command is often used to view the chapter or section heading of the statement you are reading when it's not on the screen. You will view a different portion of your file: the source plus as much of what follows as can fit. At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: Up B/A: V:

JUMP TO DOWN

Use this command to place the first substatement (one level down) of the statement you BUG at the top of the file window on your screen. You will view a different portion of your file: the substatement plus as much of what follows as can fit. At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: Down B/A: V:

JUMP TO END OF BRANCH

Use this command to place the very last statement in the branch you BUG, at the top of the file window on your screen. BUG the origin statement if you want to display the last statement in a file. You will view a different portion of your file: the end statement plus as much of what follows as can fit. At the V:, type just an OK to keep the same kind of view for the new screen; or type any Viewspecs and end with OK to change certain characteristics of your view.

C: Jump (to) B/C: End (of Branch) B/A: V:

JUMP TO RETURN

Use this command to go back to a previous view you had in your file window. After the OK, the system will prompt you with the first few words from the last screenful and "Y/N:". Type "y" for yes if you want to return to that view. If you type "n" the system will prompt you with a few words of your next-to-the-last screen view and "Y/N:". This "return ring" is a complete circle, eventually including your current view. After you have selected a return view, type OK.

C: Jump (to) B/C: Return OK: "FLASHBACK" Y/N: OK:

JUMP TO FILE RETURN

JUMP TO HEAD

JUMP TO TAIL

JUMP TO WORD

JUMP TO CONTENT

SET VIEWSPECS

Use this command to change the way you will view what's in your file window from now until you specify other changes or end this NLS work session. Some examples of the viewing certain characteristics you can control are whether blank lines appear between statements, and the number of levels and lines of statements that will appear. The V: prompts for one or more Viewspec letter codes for particular viewing feature changes. End with OK. Your screen will be re-created with the specified changes; the same statement will be at the top of the screen. Note below that you must type three characters (viz., a space, an "s", and an "e") to specify the command word "Set".

C: <SP>SEt C: Viewspecs V:

HOW TO TERMINATE YOUR WORKING SESSION

Use this command to terminate your working session.

C: <SP>Logout OK: