Table of Contents

List of Figures

Overview of the Guide

The ASSIST Software is a menu/form interface to the UNIX system. An optional menu/form generator tool, **astgen**, is also available with ASSIST. This *User's Guide* is for the menu interface only. **astgen** is described in the *ASSIST Software Release 1.0 Development Tools Guide*. The ASSIST menu interface will make using the UNIX system easier. As you become experienced with the UNIX system, ASSIST will help you to become more knowledgeable about UNIX system utilities outside your area of expertise. ASSIST helps you locate a UNIX system command and build an executable UNIX system command line. In addition, ASSIST has help messages which "pop-up" on the screen and provide information about menus, menu items, command forms, and command form items.

This guide is the primary user's documentation for ASSIST although ASSIST is intended to be self-explanatory for most users. If you want a complete explanation, read on.

You will find the following information in the guide:

- Chapter 1, "What Is ASSIST?", provides you with an overview of ASSIST and briefly describes menus, command forms, and the pop-up menu.
- Chapter 2, "Getting Started With ASSIST", gives information about step-by-step instructions to access ASSIST.
- Chapter 3, "How To Use Menus", describes ASSIST's menu hierarchy and menu format.
- Chapter 4, "How To Use Command Forms", explains how to use command forms and construct executable UNIX system command lines. It also describes the command form format, output, and automatic data validation features, and tells you how to save a command line.
- Chapter 5, "Interactive Walkthrus", tells you about the interactive simulations of UNIX system and ASSIST commands, called walkthrus, available with ASSIST and how to use them.
- Chapter 6, "Pop-up Menu", explains the pop-up menu and its function as a "home base" to the rest of ASSIST.

- Chapter 7, "Pop-up Help", gives a comprehensive picture of the pop-up help feature. It also lists and describes all the ASSIST commands.
- Appendix A is a diagram of the ASSIST menu system.
- Appendix B is the manual page for the **assist** command. It includes a command line synopsis, a description of the command, definitions of available options and option arguments, and examples.
- Appendix C explains the setup module in ASSIST.
- Appendix D is a list of the UNIX system commands supported by ASSIST.
- Appendix E is a list of ASSIST error messages with expanded explanations of the error messages produced by ASSIST.
- Appendix F is a summary of the commands covered in ASSIST's walkthrus.
- The Glossary of ASSIST Commands is a list of commands available within the **assist** program.
- The Glossary covers terms appearing in this guide, all ASSIST terms, and some UNIX system terms.
- The Index is a general index that lists the page numbers in this guide where information on subjects related to the UNIX system and ASSIST Software can be found.

A reference card and keyboard overlay (for AT&T terminals) are also included in this document's cover flap.

Typographical Conventions Used in this Guide

Type Styles Used in this Guide

Four different type styles are used in this guide:

- Palatino type, which looks like this and is used for the main body of the document.
- *Italic type*, which is used to cite references and to show that the italicized information is not to be taken literally. For example, when the word *file* appears in the command line, you type in the actual file name, not the word "file."
- **Bold type**, which is used to indicate UNIX system command names and ASSIST command names and to show what you would type at your terminal keyboard.
- `Constant width type`, which is used to show the output that you will see on your terminal screen.

Representations of the Keyboard

The following conventions are used to represent some of the terminal's keys:

- **<RETURN>** represents the carriage return key.
- **<BACKSPACE>** represents the backspace key.
- **<TAB>** represents the tab key.
- **<SPACE>** represents the space key.
- **<CTRL>** represents the control key.
- ^*letter* represents a control character in ASSIST, for example, ^A. When this notation is used, it means press the **<CTRL>** key down and strike *letter* while you hold the **<CTRL>** key down. For example, to enter the ^**D** character, hold down the **<CTRL>** key and strike the letter **d**.

- **f***x* represents a function key. Function keys are usually located on the top row of your terminal's keyboard. When this notation is used, it means press the indicated function key. For example, **f1** means press function key f1.
- **<UP-ARROW>** reperesents the arrow key that points in the upward direction.
- **<DOWN-ARROW>** represents the arrow key that points in the downward direction.
- **<RIGHT-ARROW>** represents the arrow key that points to the right.
- **<LEFT-ARROW>** represents the arrow key that points to the left.

Additional Conventions

- **$** is the UNIX system prompt. It is used in this guide to indicate that the UNIX system expects you to type a response.
- ASSIST, shown in upper-case letters, is used when reference is made to the entire ASSIST Software package, which includes the menu interface program (**assist**) and the menu/form generator program (**astgen**).
- **assist**, shown in bold, lower-case type, is used when reference is made to the **assist** program.
- **$HOME** is an environmental variable defining your HOME directory. For example, if your HOME directory is /usr/tmp, and you define **HOME=/usr/tmp**, whenever you refer to **$HOME** the UNIX system will know you mean /usr/tmp.
- A word enclosed in a box indicates a word that will be highlighted on the terminal screen.

Additional Readings

Kernighan, B. W., & Pike, R. (1984). *The UNIX Programming Environment*. Englewood Cliffs, NJ: Prentice-Hall.

ASSIST Software Release 1.0 Development Tools Guide (307-235).

UNIX System V Release 3 User's Reference Manual (307-232).

UNIX System V Release 3 Programmer's Reference Manual (307-226).

Chapter 1: What Is ASSIST?

Introduction

ASSIST is a menu interface that will make using the UNIX system easier for you. It helps you get started with the UNIX system, categorizes UNIX system commands by function, and helps you build and execute commands.

ASSIST will help you to become more knowledgeable about UNIX system commands outside your area of expertise.

Unlike some menu interfaces to the UNIX system, ASSIST does not hide the UNIX system from you. Instead, it is a *learn-while-doing* environment. ASSIST consists of:

- Menus
- Command forms
- Pop-up menu

Full-Screen Menus

ASSIST menus are conceptually similar to a restaurant's menu. You look at a list of possible choices, then make your selection. The menu system helps you find the UNIX system command for the function you want to perform. At the same time, the menus show the range of functionality available from the UNIX system. ASSIST menus are organized in a hierarchy with the TOP menu at the root, followed by menus that list UNIX system commands and command forms at the tips of the hierarchy. (Appendix A diagrams this hierarchy.)

ASSIST menus feature:

- Pop-up help that is accessible from any point in ASSIST
- Easy return to previous menus
- Easy return to the UNIX system shell

ASSIST menus can be accessed in three ways:

- By typing **assist** from your UNIX system prompt
- From other menus
- From command forms

Command Forms

Command forms are interactive forms that help you run UNIX system commands. Filling out a command form is similar to filling out a paper form. Both have printed text and blank spaces for insertion of required or requested information. As you fill out the command form, ASSIST validates each field and your responses are recorded on the form. At the same time, a syntactically-correct command line reflecting your valid input up to that point is displayed at the bottom of the screen. If you give an invalid response, an error message or a warning message is given. After using a command form a few times to run a specific command, you will have enough experience with that command to execute it directly from the UNIX system command level. Also, you can use a command form to create and store command lines for commands that are difficult to use, or for commands that you use infrequently.

A command form is a full-screen form containing a short description of each option and argument that can be used with the command. ASSIST command forms feature:

- Validation of user input
- Automatic command line generation
- Pop-up help that can be accessed from any point in ASSIST
- Visible shell expansion. For example, **$HOME** is expanded to your full pathname on the screen.

ASSIST command forms can be accessed in three ways:

- By typing **assist** from your UNIX system prompt
- From ASSIST's full-screen menus
- From the ASSIST Pop-up Menu

Pop-up Menu

The pop-up menu serves as a "home base" that you can invoke from ASSIST's menus and command forms. It provides you with direct access to ASSIST's features, directory switching, a keyword-based command search capability, and introductory walkthrus about the UNIX system and ASSIST.

Chapter 2: Getting Started with ASSIST

Invoking ASSIST

After you have logged in on the computer, the shell prompt **$** appears. To access ASSIST, you must type the **assist** command after the prompt. When you type **assist** the command line looks like this

```
$ assist
```

(The manual page is shown in Appendix B. It provides a command line synopsis, and a description of the command and its options.)

Normally, when you access ASSIST, the TOP menu appears on your screen as shown in Figure 2-1. However, if you are using ASSIST for the first time or if your terminal is incorrectly defined, ASSIST helps you get started by doing some preliminary checks and verifications. For a description of this setup procedure, see Appendix C.

Cursor Movement

Use either ^P or the **<UP-ARROW>** key to move the cursor up. The cursor is indicated by the greater-than sign (>) and shows your current location on the screen. Use either the **<RETURN>** key or the **<DOWN-ARROW>** key to move the cursor down.

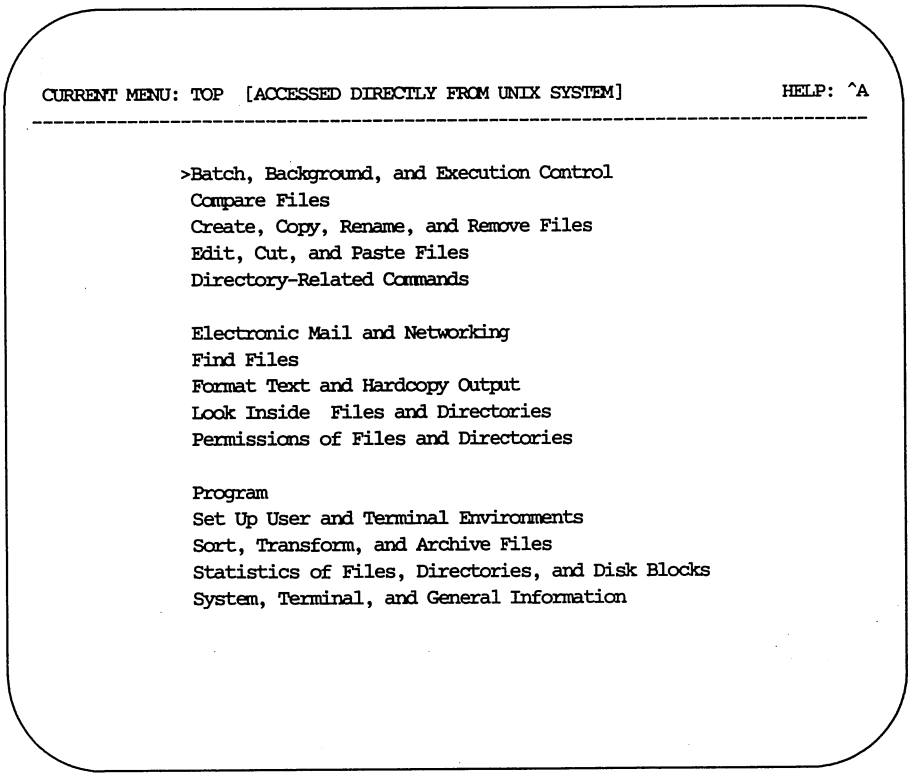


Figure 2-1: TOP Menu

On menus, you can also go directly to an item by striking the first letter of the desired item. For example, in Figure 2-1, in order to move to the Permissions of Files and Directories menu, you can type **p**. If there is more than one entry beginning with the specified letter, the cursor moves to the first item.
On the TOP menu shown in Figure 2-1, there are two items beginning with P. The cursor moves to Permissions of Files and Directories. If you type **p** again, it will move down to Program.

The keys used to select items and fields are **f1** or **^G**.
Strike either **f1** or **^G** to select the Program menu from the TOP menu shown in Figure 2-1.

Chapter 3: ASSIST's Full-Screen Menus

Introduction

ASSIST's menus provide organized access to UNIX system commands. The menus are categorized according to UNIX system command functions. For example, commands relating to formatting text and hardcopy output are on one menu and commands relating to programming are on another menu. The TOP menu is the first-level menu. There are fifteen second-level menus that are accessible from the TOP menu. A few of these menus have third-level menus (see Appendix A).

You have already learned in Chapter 2, how to access ASSIST, obtain the TOP menu, move the cursor, and select items. You can select any menu on an ASSIST menu by striking either **f1** or **^G**. You can return to a previous menu from the current menu by striking either **f4** or **^R**. You can always return to the TOP menu by striking **^T**.

How Menus Are Organized

ASSIST menus have the following features:

- Header Line
- Selectable Items
- Message Line

Figure 3-1 is an example of a typical ASSIST menu.

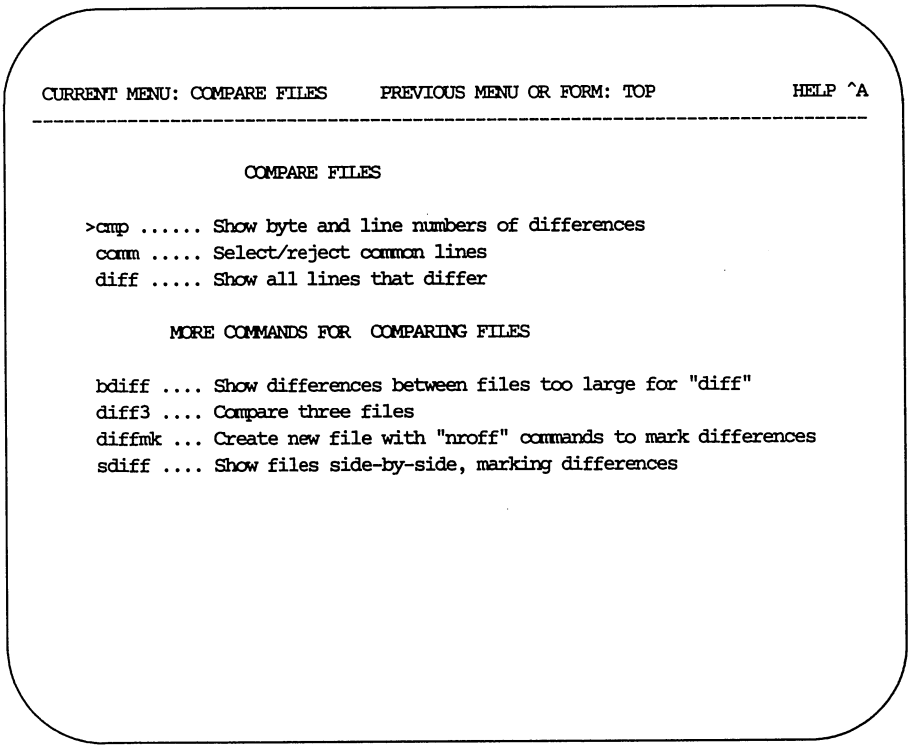


Figure 3-1: Compare Files Menu

Header Line

The header line shows the name of the current menu and the name of the previous form. In Figure 3-1, the header line tells you that the current menu is the COMPARE FILES menu, and that you accessed this menu from the TOP menu. This line also indicates that you can get help by using the ^A control character. When you hit ^A, ASSIST gives you a summary of ASSIST commands. (For information about menu pop-up help, see Chapter 7.)

Selectable Items

Below the top line are the selectable items. The items can either be a list of other menus or a list of command forms. The current item is indicated by a "greater than" sign, (>), at the beginning of the line. If your terminal has highlighting, the item is highlighted too.

In Figure 3-1, the selectable items are names of command forms. The list is divided into two groups. The first group consists of the command forms that are currently available through ASSIST. The second group listed under MORE COMMANDS FOR COMPARING FILES contains other commands associated with the menu category. These command forms are not currently available through ASSIST, but may be in future releases. You can, however, ask for a brief description of the commands by striking ^Y after moving to the item. The items listed under MORE COMMANDS ... thus function as "pointers" to other UNIX system commands you may want to try on your own.

Message Line

The message line is at the bottom of the menu. A message appears here when an inappropriate command is used or when a command form in the MORE COMMANDS FOR <*menu name*> section is chosen. There is no message line shown in this example.

Chapter 4: ASSIST's Command Forms

Introduction

ASSIST's command forms are interactive forms that help you to execute UNIX system commands. The command forms give you a friendly environment for trying new commands. They also help you to understand how the commands work. They tell you the command's function, the meaning of its options, and which command line arguments are valid. Command forms help you understand the proper command line syntax. They generate command lines for you. Then you can execute and store command lines for commands that are difficult to use.

Appendix D lists the UNIX system commands supported by ASSIST command forms.

Command Form Layout

The ASSIST command form has the following features:

- Header Line
- Option and Argument Specification Fields
- Command Line
- Error Message Line

Figures 4-1 and 4-2 show the **ls** command form after it has been filled out.

```
CURRENT FORM: ls   PREVIOUS MENU OR FORM: ls menu                    HELP: ^A
-------------------------------------------------------------------------------

     NAME OF INPUT FILE(S)/DIRECTORY(S):
     LIST NAMES BEGINNING WITH "." [-a] (y/n): y
     RECURSIVELY LIST SUBDIRECTORIES ENCOUNTERED [-R] (y/n): y
     LIST DIRECTORY NAME ONLY, NOT ITS CONTENTS [-d] (y/n): n

   ATTRIBUTE LIST
     LIST IN LONG FORMAT [-l] (y/n): y
     LIST IN LONG FORMAT, PRINT UID AND GID NUMBERS [-n] (y/n): n
     LIST IN LONG FORMAT, BUT DO NOT PRINT GROUP [-o] (y/n): n
     LIST IN LONG FORMAT, BUT DO NOT PRINT OWNER [-g] (y/n): n

   SORTING OPTIONS
     REVERSE ALPHABETIC OR OLDEST FIRST [-r] (y/n): y
     TIME MODIFIED [-t] (y/n): y
     TIME OF LAST ACCESS [-u] (y/n): n
     TIME OF LAST i-NODE MODIFICATION [-c] (y/n): n

-------------------------------------------------------------------------PAGE 1 OF 2
 COMMAND LINE: ls -aRlrt > temp
-------------------------------------------------------------------------------
```

Figure 4-1: **ls** Command Form (Page 1)

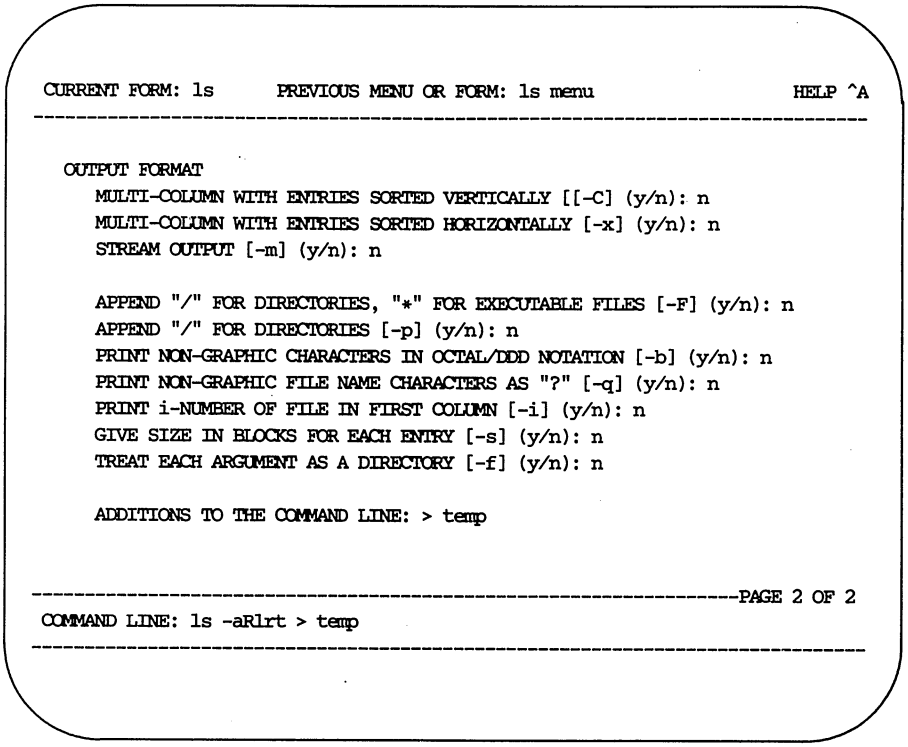


Figure 4-2: **ls** Command Form (Page 2)

Header Line

The header line is the same as the header line of ASSIST's menus. This line tells you what command form you are currently using. It also indicates where you were before you got to this command form. In this example, the current form is the **ls** command form and it was accessed directly from the ls menu. This line also indicates how to get help, by using the **^A** control character. When you hit **^A** or **f8**, ASSIST gives you a summary of the ASSIST commands pertaining to command forms. For information about ASSIST commands and command form pop-up help, see Chapter 7.

Option and Argument Specification Fields

These descriptive fields make up the main body of the command forms. On command forms, the fields describe the UNIX system command arguments and options. A field consists of a description, the name of the option in brackets when applicable, and sometimes a (y/n) indicating that this field requires a yes/no answer. As you can see in Figure 4-1, the first field requires that you type in the name of a file or directory. All the other fields in this example require a yes/no answer.

In long forms, the fields may be categorized according to similarity of function of the options. For example, in Figure 4-1, the field groups are named ATTRIBUTE LIST and SORTING OPTIONS.

If the command form has more than one page, ASSIST tells you what page you are on and the total number of pages in the command form. This information appears on the right side of the form near the bottom. On the **ls** command form in Figure 4-1, you can see that you are on the first page of a two-page command form. Strike **^U** when you want to turn to the next page.

The last field in this section is labelled ADDITIONS TO THE COMMAND LINE. This is shown in the continuation of the **ls** command form in Figure 4-2. Enter any additonal data here, such as piping or redirection commands. ASSIST does not check your input to this field, so use it with care.

Command Line

The command line is on the next to the last line on the screen. It contains the UNIX system command line that corresponds to what you have typed in. Each time you add a piece of data to the command form, the command line is updated to reflect the new information. As an executable command line is being built, it appears on your screen. Command forms not only assist you in executing a UNIX system command, they also teach you. When you have successfully completed a command form, you have learned how to construct a syntactically-correct command line.

Error Message Line

When you make an incorrect entry on a command form, the error message line either prints a left-adjusted error message or a right-adjusted warning message. Error messages are given for entries which will cause command execution to fail. Warning messages tell you of potential trouble, for example, that you will overwrite an existing file. ASSIST will tell you about this possibility and let you decide if you want to overwrite the file. (Appendix E contains an expanded explanation of many of the ASSIST error messages.)

Accessing Command Forms From the ASSIST Menus

The following example shows you how to access the **rm** command form by traversing the menu hierarchy. Type the following command:

```
$ assist
```

When you type this, the TOP menu appears on your screen as shown in Figure 4-3.

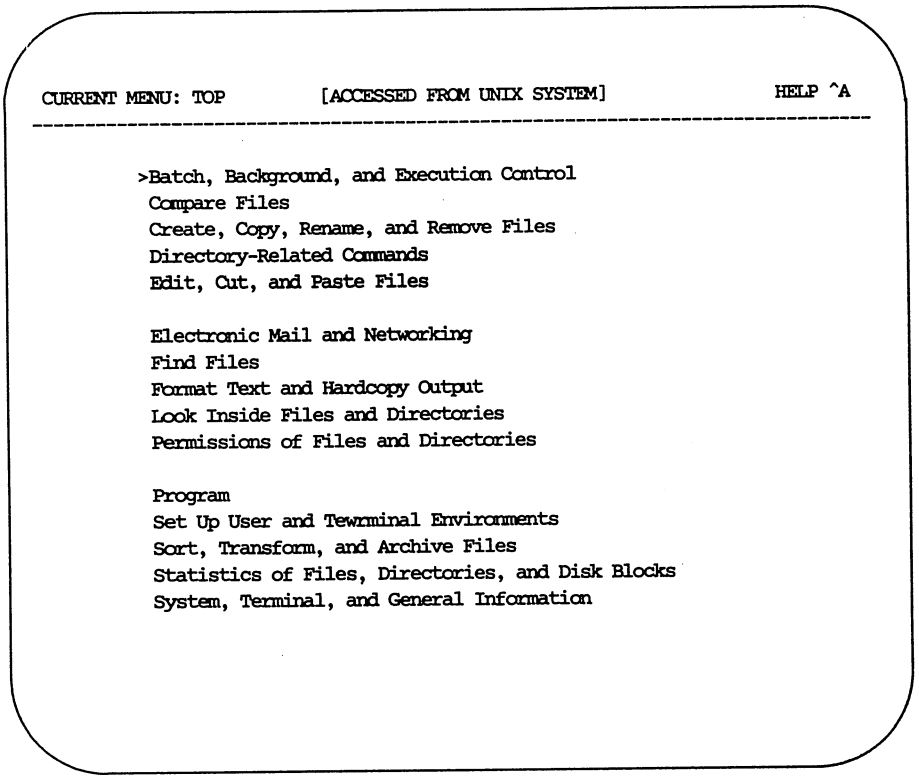


Figure 4-3: ASSIST's TOP Menu

Move the cursor down to the Create, Copy, Rename, and Remove Files item by striking <DOWN-ARROW> twice or striking <RETURN> twice. Select this item by striking either **f1** or ^G. The menu appears on your screen as shown in Figure 4-4.

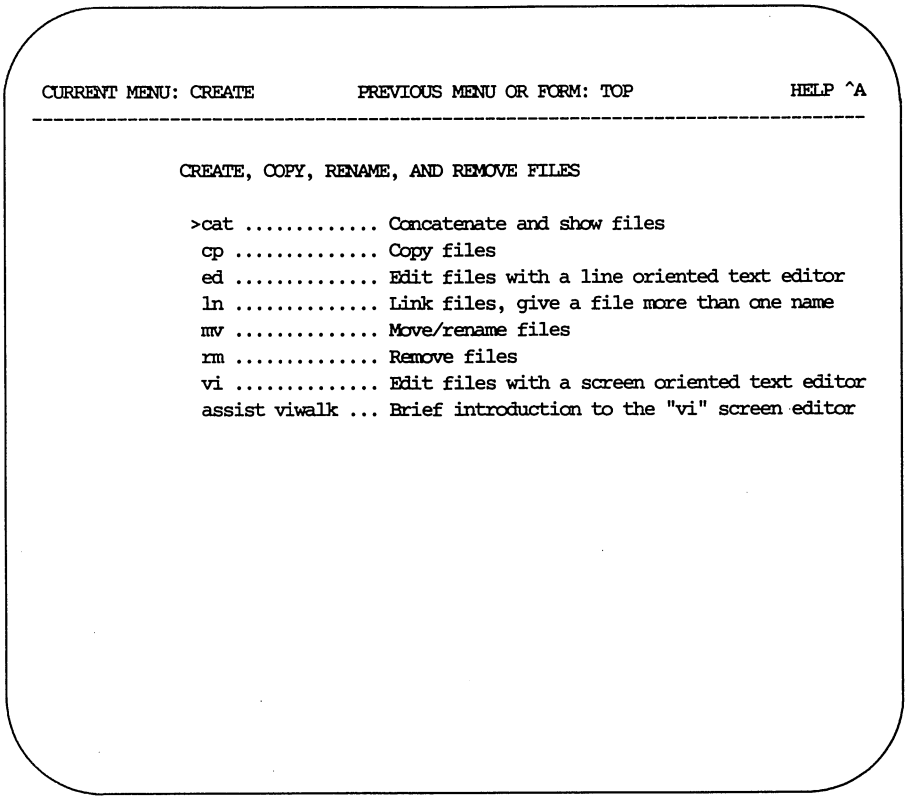


Figure 4-4: Create, Copy, Rename, and Remove Files Menu

The first seven items on this menu are the names of UNIX system commands. The last item assist viwalk is the command you would type to access the **vi** text editor walkthru.

Move the cursor down to the **rm** item by striking either **<DOWN-ARROW>** or **<RETURN>** five times. Select this item by striking either **f1** or **^G**. The **rm** command form will appear on your screen as shown in Figure 4-5. Now you may proceed to fill out the **rm** command form. Instructions for filling out a command form are given in the section called "How To Fill Out a Command Form."

```
CURRENT FORM: rm          PREVIOUS MENU OR FORM: CREATE                HELP ^A
-------------------------------------------------------------------------------

     NAME OF FILE(S):

     REMOVE DIRECTORIES AND THE CONTENTS OF THE DIRECTORIES [-r] (y/n): n

     NAME OF DIRECTORY(S):

  OPTIONS FOR FILES AND DIRECTORIES
     PROMPT FOR FILE(S) WITH INCORRECT PERMISSIONS [-f] (y/n): y
     PROMPT BEFORE REMOVING EACH FILE AND/OR DIRECTORY [-i] (y/n): n

     ADDITIONS TO THE COMMAND LINE:




-------------------------------------------------------------------------------
 COMMAND LINE: rm
-------------------------------------------------------------------------------
```

Figure 4-5: **rm** Command Form

After you have used ASSIST for some time, you will have enough experience with UNIX system commands to bypass the menus altogether. You can go directly to the command form you want to use by typing

$ assist *commandname*

For example, if you want to use the **rm** command form, type

$ assist rm

This takes you directly to the **rm** command form.

How To Fill Out a Command Form

How To Move Between Fields

Moving between fields is done with:

- <RETURN>
- <DOWN-ARROW>
- <UP-ARROW>
- ^P

To move to the next field, strike either <RETURN> or <DOWN-ARROW>. To move up to a previous field, strike either ^P or <UP-ARROW>. If you are on the last field and strike <RETURN> or <DOWN-ARROW>, the cursor moves to the first field. This "wrap" feature also applies to upward movement. If you are on the first field and strike <UP-ARROW> or ^P, the cursor moves to the last field.

How To Enter Input

There are typically three ways to enter input in a field:

- Type a single character
- Type a single word
- Type several words

To enter a single character, simply type the character and strike <RETURN>. To enter a single word, type the word and strike <RETURN>. To enter more than one word, type the words separated by spaces and strike <RETURN>.

How To Move Left and Right

Moving left and right is done with:

- <LEFT-ARROW>
- ^B
- <RIGHT-ARROW>
- ^W

Some fields allow multiple, space-separated entries. To move left, strike either **^B** or **<LEFT-ARROW>**. To move to the next entry to the right, strike either **^W** or **<RIGHT-ARROW**. When you move to the left or right, you can move between the entries you have made on a line, but you cannot move from letter-to-letter in a particular entry. For example, if you type **file1 file2**, you can move from the word **file1** to the word **file2**, but you cannot move to the individual letters in the words **file1** and **file2**.

How To Enter Special Characters

Any printable character is included and displayed in the input area. ASSIST commands such as **^G** and shell meta characters such as >, <, |, *, and ", are examples of special characters. The ASSIST command, **^C**, is used on command forms to escape characters that have special meaning in ASSIST. For example, if you want to type **^B** on a command form, you must first strike **^C**, then strike **^B**.

ASSIST provides shell expansion for characters that perform functions but are not printable themselves. When you enter one of these characters, they are represented on the command line by their ASCII equivalent. For example, **<TAB>** is shown as TAB.

How To Edit a Field

You can return to a field to edit it by using **<UP-ARROW>** and **<DOWN-ARROW>**. If you enter an input area that already contains input, typing a printable character results in the entire word or character being deleted. The new character is displayed in the input area. Striking **<SPACE>** has the same effect; one word is removed for every **<SPACE>** you strike. **<SPACE>** also indicates the end of a word.

Some fields have a value that is automatically supplied by ASSIST. This is the default value and is in effect unless you change it. When you strike **<SPACE>**, ASSIST restores the default value, if there is one. If there is no default, the previous input for this field is erased and the field is left blank.

To erase characters, strike **<BACKSPACE>** for each character that you want to erase.

Data Validation

When ASSIST validates the command line, it checks for errors that would result in a UNIX system error message. This includes checking for file existence, read permission, and command line syntax.

Data validation is done each time a new piece of data is added to a field or when you try to execute the command line. You may try to execute the displayed command line at any time. When ASSIST validates, a message may flash on the lower right corner of your screen. The message looks like this:

```
VALIDATING INPUT, PLEASE WAIT ...
```

This message occurs when there is a time-consuming validation or when you leave a field. The command line will execute if all required arguments are present. ASSIST will not execute the command if something is missing or has been incorrectly specified. Input validation takes place when you:

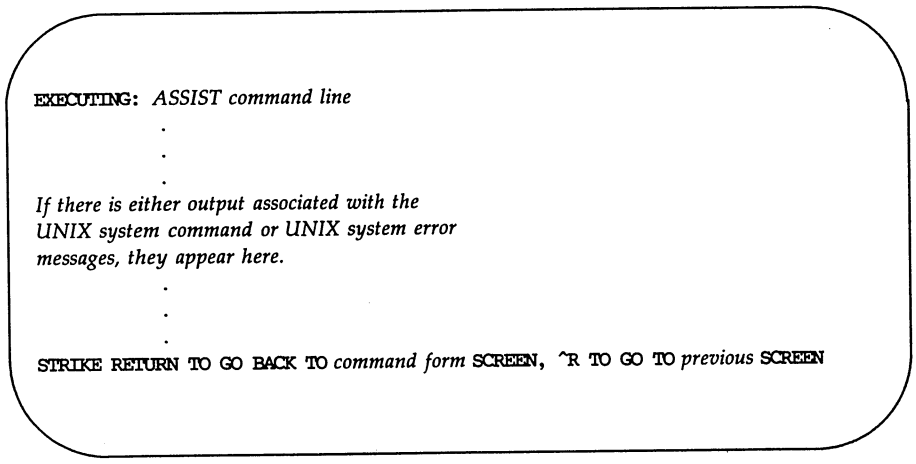
- Attempt to leave the current input area
- Strike **<SPACE>**, unless the function of **<SPACE>** is turned off by using quote characters or backslashes
- Try to execute the command line
- Store the command line

Here's a word of caution about validating fields. If you have a field with several incorrect entries, you will get an error message. However, if when you edit that field to correct your errors, you do not change all of the incorrect entries, ASSIST does not reissue the error message. ASSIST assumes that all corrections have been made.

How To Execute an ASSIST Command Line

After you have filled out a command form, it is ready to be executed. Strike **f1** or **^G** to execute the command line. When you do this, the validation message may flash on the lower right corner of your screen.

Some ASSIST command forms have execute messages. Execute messages tell you about any special actions you must take to execute the command. Typically, these messages appear when you are about to run a UNIX system command that requires input from your terminal. As soon as the input is validated, the execute message appears, if there is one. Otherwise, the command form disappears from the screen and the following text appears:



ASSIST has executed the command. At this point, you can either return to the same command form by hitting **<RETURN>** or the previous screen by hitting **f4** or **^R**. If you entered the command form directly from the UNIX system, **f4** or **^R** returns you to the UNIX system prompt.

How To Save a Command Line

You may want to save a command line. For example, you may be using a difficult UNIX system command with many options. Suppose you have completed the command form for the **sort** command and the command line looks like this:

```
COMMAND LINE: sort -rb +1.0 -1.1 infile1 infile2
```

Strike ^K to save the command line. The following message appears on the bottom line in the right corner:

```
COMMAND LINE WRITTEN IN FILE sort.assist
```

ASSIST has saved the command line in file **sort.assist** in your current directory with execute permissions. (The file, **sort.assist**, containing your stored command line, will be overwritten the next time you store a new command line for **sort** in this directory. Therefore, if you want to save this file, you should copy or move it, giving it a different name.) When you want to execute the saved command line in the future, you simply type the name of the file where the line is stored. In this case, type **sort.assist**.

Chapter 5: ASSIST's Walkthrus

Introduction

The ASSIST walkthrus are interactive simulations of UNIX system commands or concepts. Walkthrus provide introductory information designed to allow you to begin using the command or concept being taught or "walked through." ASSIST has four walkthrus:

- ASSIST
- **sdb** (a debugger for C programs)
- UNIX system
- **vi**

ASSIST provides an introductory tutorial designed to walk you through the basic concepts and commands that ASSIST uses. We strongly recommend that you use this walkthru. You will only be told about this by the setup module described in Appendix C when you are a new user.

How To Access Walkthrus

The introductory ASSIST and UNIX system walkthrus can be accessed from ASSIST's pop-up menu and from the `System`, `Terminal`, and `General Information` submenus.

The **vi** and **sdb** walkthrus can be accessed from the `Edit`, `Cut`, and `Paste Files` menu and the `Debug` submenu of the `Program` menu, respectively. You can also access all of the walkthrus by typing

```
$ assist assistwalk
$ assist sdbwalk
$ assist unixwalk
$ assist viwalk
```

How To Use Walkthru Commands

When you are in a walkthru, you can type ^A or **f8** to get a list of ASSIST commands that apply to walkthrus. The only way to use an ASSIST command in a walkthru is to first strike ^A or **f8**. Then strike the walkthru command. Figure 5-1 shows you these commands.

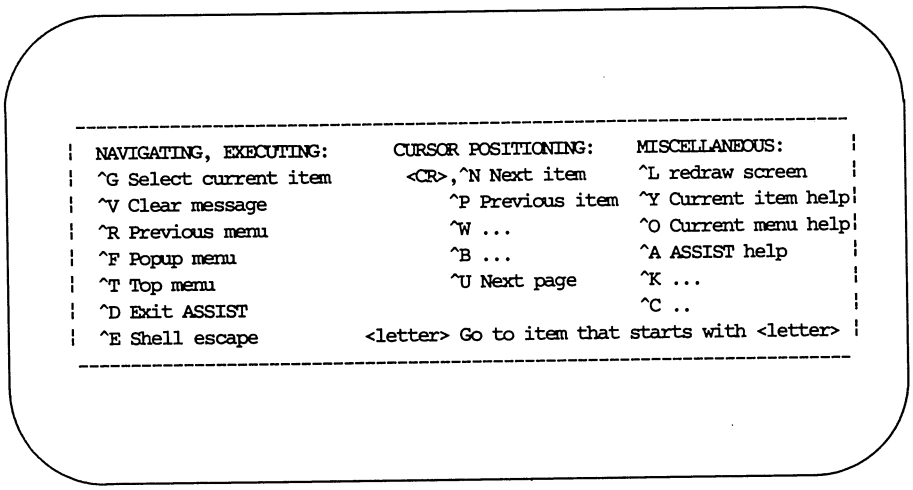


Figure 5-1: Walkthru Commands

How To Exit Walkthrus

You can exit a walkthru by striking **^A** or **f8**, then striking:

- **^R** or **f4** to return to the previous menu
- **^T** to return to the TOP menu
- **^D** to exit **assist**

See Appendix F for a summary of commands covered in the walkthrus.

Chapter 6: ASSIST Pop-up Menu

Introduction

The pop-up menu provides access to many of ASSIST's functions. You can access command forms and two of ASSIST's walkthrus, the ASSIST walkthru and the UNIX System walkthru. The pop-up menu, like the full-screen menus, also has pop-up help. For a description of the pop-up help, see Chapter 7.

How To Access the Pop-up Menu

The pop-up menu can be accessed from either a menu or a command form with a single keystroke. Type **f5** or ^F to call the pop-up menu. Figure 6-1 shows the Electronic Mail and Networking menu with the pop-up menu overlaid.

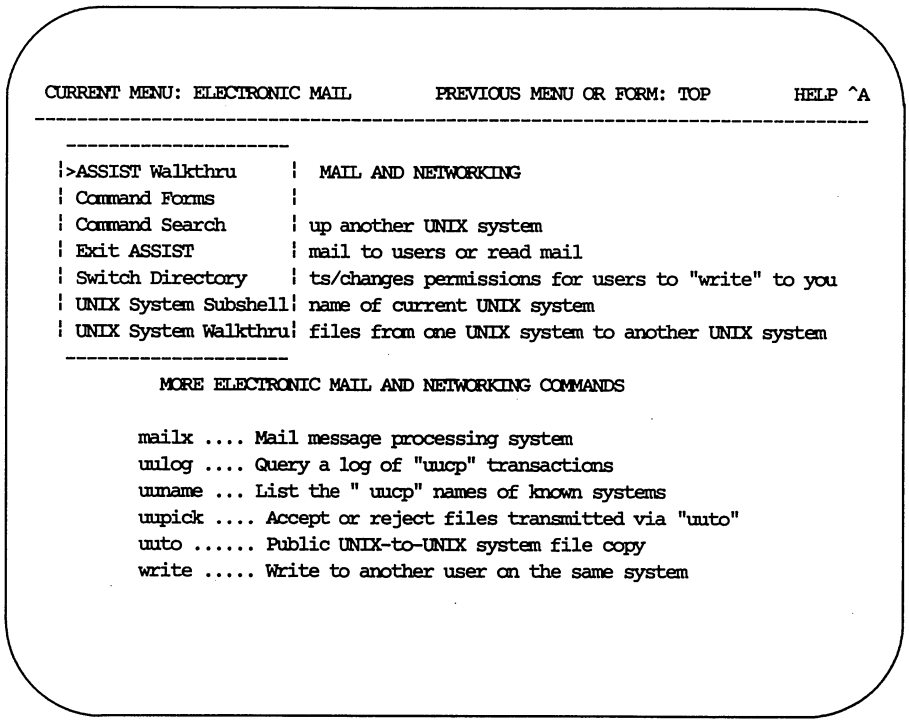


Figure 6-1: Electronic Mail and Networking Menu With Pop-up Menu

After you have positioned the cursor on the pop-up menu item that you want to select, strike **f1** or ^G.

Cursor Movement

The cursor movement conventions are the same as those for full-screen menus. Upward movement occurs when you type **^P** or the **<UP-ARROW>** key. Downward movement occurs when you type **<RETURN>** or the **<DOWN-ARROW>** key. The "first letter" method also works on the pop-up menu. Simply type the first letter of the desired item and the cursor moves to that item. Select with **f1** or **^G**.

How To Exit the Pop-up Menu

There may be occasions when you choose the pop-up menu and then decide you really did not want to use it. Strike ^**R** or **f4** to exit the pop-up menu. This causes it to disappear from the screen and returns you to the previous menu or command form.

Pop-up Menu Items

The pop-up menu contains seven items. Each item is described in the following sections.

ASSIST Walkthru

The `ASSIST Walkthru` offers brief, interactive demonstrations of the key concepts and commands of ASSIST. When you choose this item, a menu appears. You can choose the complete walkthru which covers all of ASSIST, or you can choose a shorter walkthru on a particular aspect of ASSIST. As a new ASSIST user, you should do the complete walkthru. Later, you can do sections to refresh your memory.

Command Forms

The pop-up menu eliminates the need to use the full-screen menus to select command forms. When you select `Command Forms`, a prompt appears on the bottom of the screen. The prompt gives you two choices:

- Type the name of the command form.
- Choose the menu that tells you what command forms are available.

If you know which command form you want to use, type the command name. For example, to get the **rmdir** command form, type **rmdir**.

If you name a command form that does not exist, the following message appears at the bottom of the screen:

<*command*> `NOT SUPPORTED BY ASSIST [TYPE RETURN]:`

When you strike **<CR>**, the command form prompt reappears on the screen.

If, when the prompt appears, you do not know what command forms exist or are not sure of the appropriate one to use, strike **<RETURN>** and a menu appears. This menu contains an alphabetical list of all of the command forms and a brief description of the UNIX system command function.

To exit the Command Forms menu, strike ^R or f4. When you exit the Command Forms menu, you return to the menu or form from which you called the pop-up menu.

Command Search

There may be occasions when you want to perform a function but do not know what command to use. Or, you may remember using a command but do not remember its name. To find it, you can use ASSIST's menus or the Command Search feature. The Command Search will help you find the name of the appropriate command. It prompts you for a *search-word*. Then ASSIST scans the database containing descriptions of the UNIX system commands contained in Section 1 of the *UNIX System V Release 3.0 User's Reference Manual* and their options for the *search-word* or a synonym of that word. All commands that have the *search-word* in their description are listed. The *search-word* should be as unique to the command as possible. For example, if you want to remove a file, you could use either **remove** or **file** as the *search-word*. If you use **file** as the *search-word*, you will get a very long list of commands. The word **file** is too general. The word **remove** is much more specific and the list will be much shorter.

When you select the Command Search item, the following prompt appears on the bottom of the screen:

```
TYPE SEARCH WORD:
```

Let's use the example mentioned above, removing a file. When you type **remove**, all of the commands containing **remove** in their description are listed. If there is more than one page of output from the search, strike **<RETURN>** to continue the search.

After all of the appropriate commands have been listed, the prompt at the bottom of the screen looks like this

```
STRIKE RETURN TO CONTINUE WITH MENU, OR TYPE COMMAND NAME:
```

If you find the command you want to use, type in the name of the command. The command form appears on the screen. Fill it out as you would any other command form. If you want to go back to the previous menu, strike **<RETURN>** and the previous menu appears on the screen.

Exit ASSIST

Selecting this item has the same effect as typing ^D. It puts you back at the UNIX system command level.

Switch Directory

There may be times when you are using ASSIST and suddenly discover that you are in the wrong directory. To change directories while you are in ASSIST, you can use the Switch Directory item on the pop-up menu. Then the directory will be changed until you exit ASSIST or reissue the command. When you choose Switch Directory, the following prompt appears at the bottom of the screen:

CURRENT DIRECTORY: <*current directory name*> NEW DIRECTORY:

This prompt tells you the name of the current working directory, and prompts you for the name of the directory to which you want to switch. Type the new directory name. If the directory specified is a valid one, the following message appears

MOVED TO: <*new directory name*> [TYPE RETURN]:

If an invalid directory name is typed the following error message appears at the bottom of the screen.

DIRECTORY DOES NOT EXIST OR IS NOT ACCESSIBLE [TYPE RETURN]:

At this point, you are prompted again for a new directory name. If your current directory has a long name, you will only see the NEW DIRECTORY: prompt.

UNIX System Subshell

The UNIX System Subshell temporarily takes you out of ASSIST and puts you at the UNIX system command level. The screen is cleared and the system prompt, <ASSIST>, appears. The UNIX System Subshell allows you to interact with the UNIX system directly. When you have finished the UNIX system tasks, strike ^D to return to ASSIST. When you re-enter ASSIST, you will be returned to the same spot you were in when you left ASSIST.

UNIX System Walkthru

The `UNIX System Walkthru` provides brief, interactive demonstrations of the key concepts and commands of the UNIX system. When you choose this item, a menu appears. The menu contains a list of items concerning files, directories, input, output, and redirection.

Chapter 7: Pop-up Help

Introduction

Help is only a keystroke away in ASSIST. There are three types of pop-up help available:

- ASSIST Commands Help
- Menu/Form Help
- Item Help

When you ask for help, a box containing the information "pops up" onto the screen. These messages give you further information about the item you selected.

ASSIST Commands Help

The ASSIST commands help gives a list of the available ASSIST commands for the current screen. It tells you how to move between screens, move within screens, select help, redraw the screen, and save a command line.

How To Access ASSIST Commands Help

You can select the ASSIST commands help by striking **f8** or **^A**. The commands that are appropriate for your current position in ASSIST appear on the screen. For example, if you are in a menu, the commands that apply to menus appear on the screen. Figure 7-1 is an example of the ASSIST commands help for menus.

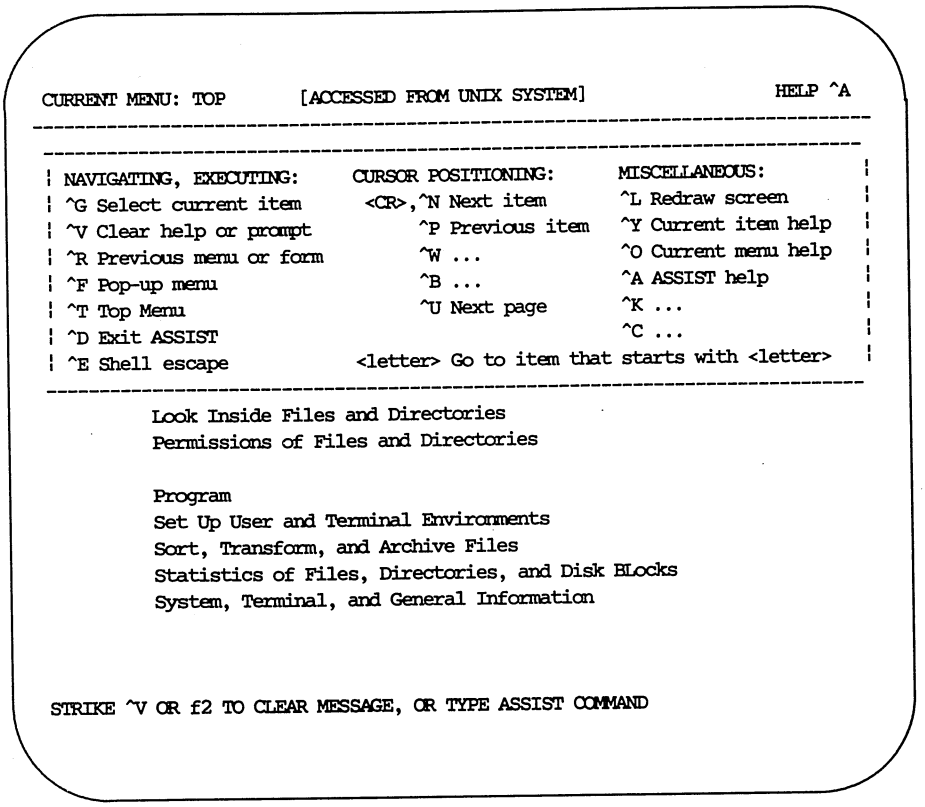


Figure 7-1: ASSIST Commands Help for Menus

Figure 7-2 is an example of the ASSIST commands help for command forms. Figures 7-1 and 7-2 illustrate the differences between the pop-up help for menus and command forms. Most commands are the same for menus and command forms.

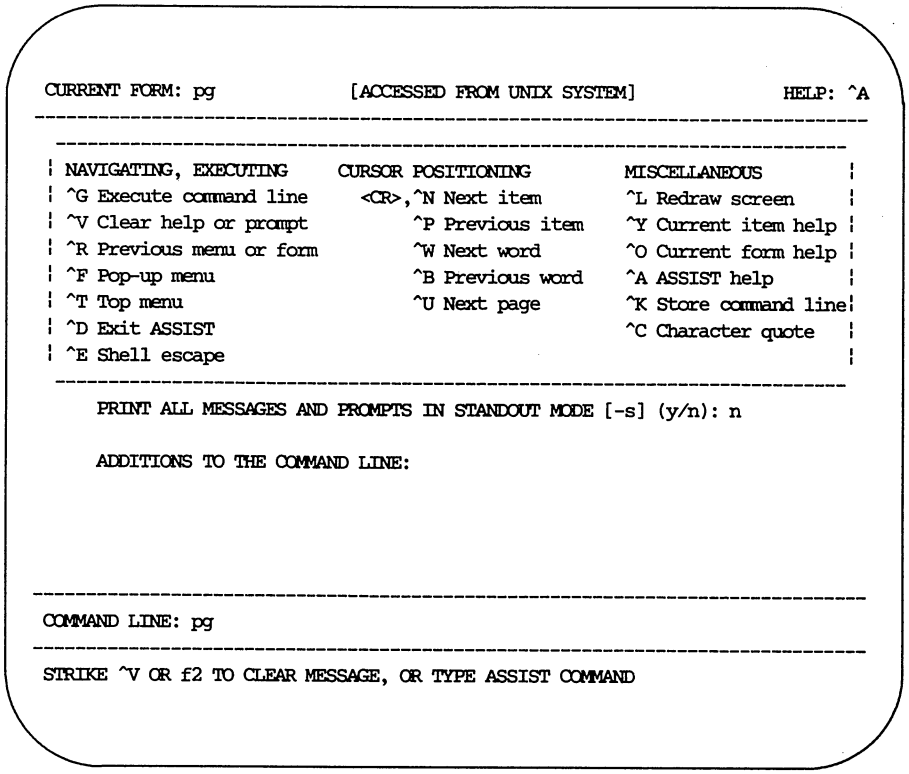


Figure 7-2: ASSIST Commands Help for Command Forms

How To Exit ASSIST Commands Help

You can erase ASSIST help by striking **f2**, **^V**, or by typing any ASSIST command. For example, if you use the pop-up help to find out how to access the pop-up menu, you will see that **^F** calls the pop-up menu. If you strike **^F**, the help message will disappear and the pop-up menu will appear with one keystroke.

The Glossary of ASSIST Commands contains a complete list and description of the ASSIST commands.

Menu/Form Help

The menu/form help gives you information about the current menu or form. If you are on a menu, the help tells you about the scope of the menu and gives some examples of the commands listed on that menu. If you are in a command form, the help describes the function of the UNIX system command.

How To Access Menu/Forms Help

To get the menu/form pop-up help, strike **^O** or **f7**. For example, if you are on the `Compare Files` menu, when you strike **^O** or **f**, the pop-up help message looks like Figure 7-3.

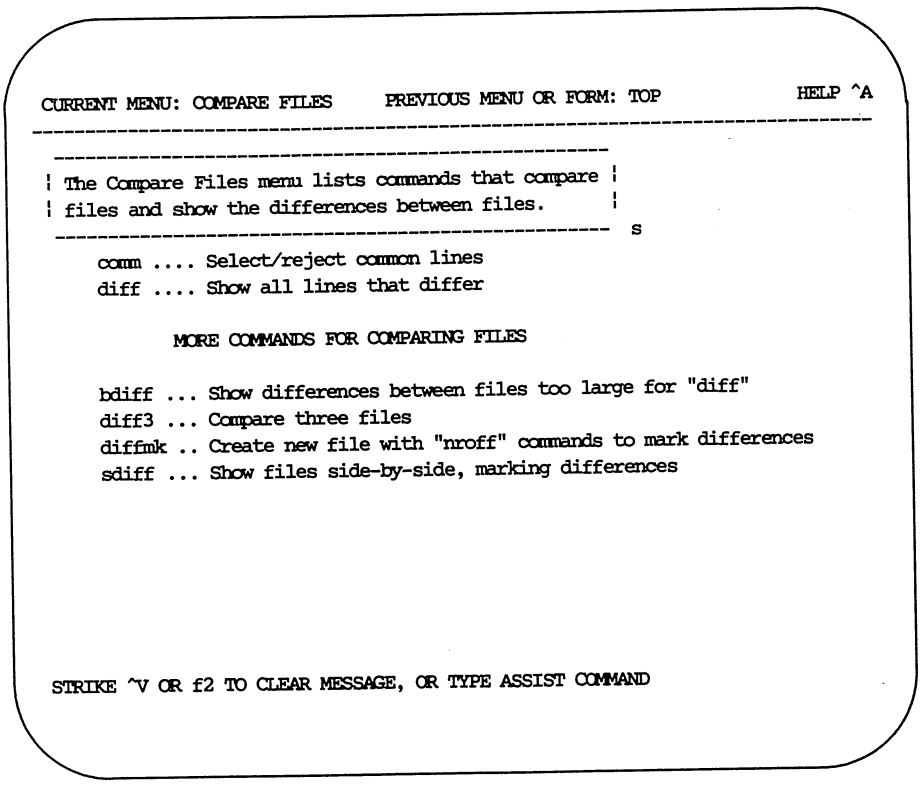


Figure 7-3: Example Menu Help Message

If you are in a command form, such as the **cmp** command form, the pop-up help message, obtained when you strike **^O** or **f7**, is the one shown in Figure 7-4.

```
CURRENT FORM: cmp       PREVIOUS MENU OR FORM: COMPARE FILES          HELP ^A
-------------------------------------------------------------------------------
 ---------------------------------------------------
 ¦ "cmp" is used to compare two files.  By default,  ¦
 ¦ "cmp" produces no output if the files are identical.¦
 ¦ If they differ, "cmp" lists the byte and line number¦
 ¦ of the first difference.                         ¦
 ---------------------------------------------------

     LIST ALL DIFFERENCES [-l] (y/n): n

     RETURN ONLY EXIT CODE [-s] (y/n): n

     ADDITIONS TO COMMAND LINE:




-------------------------------------------------------------------------------
COMMAND LINE: cmp
-------------------------------------------------------------------------------
STRIKE ^V OR f2 TO CLEAR MESSAGE, OR TYPE ASSIST COMMAND
```

Figure 7-4: Example Command Form Help Message

How To Exit Menu/Form Help

You erase the menu/form help in the same way you erase the ASSIST commands help by striking **f2**, **^V**, or by typing any ASSIST command.

Item/Field Help

How To Access Item/Field Help

The pop-up help for an item on a menu or field on a form gives you information about the current item or field. You can get pop-up help by striking **^Y** or **f6**.

If you are on a menu, the current item is usually a UNIX system command name. The help message for this item describes the function of the UNIX system command. Occasionally, the items on a menu will be the names of sub-menus. In this case, the help message gives you information about the scope of the sub-menu and some examples of commands listed on that menu.

Figure 7-5 is an example of pop-up help for menu items. The menu is the `Compare Files` menu and the item is the **comm** command.

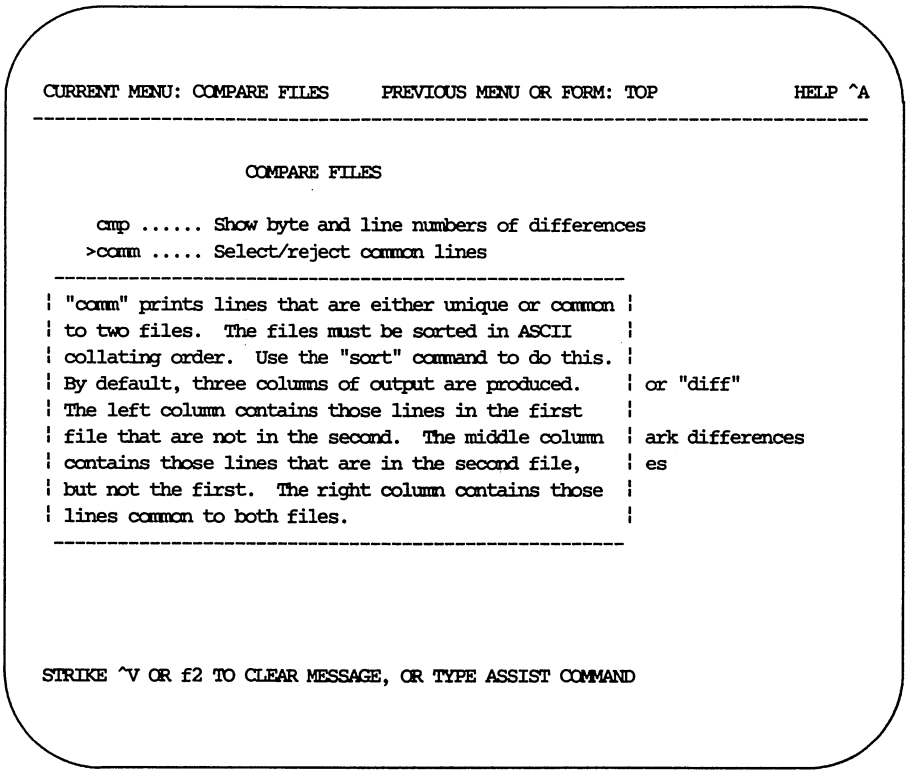


Figure 7-5: Compare Files Menu With Item Help

If you are in a command form, help for the current field describes the function of a command option or argument. It also tells you what options, if any, are incompatible with the current item. Figure 7-6 is an example of pop-up help for a field on a command form. The command form is the **dircmp** command form. The field in question is

```
SHOW CHANGES NEEDED TO BRING FILES INTO AGREEMENT (y/n): n.
```

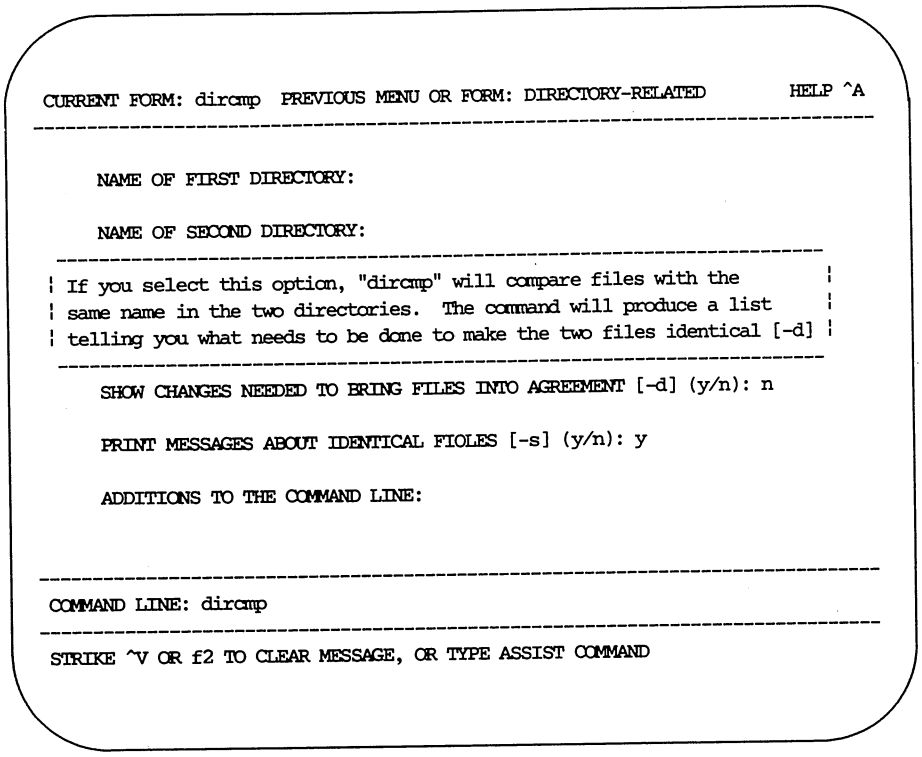
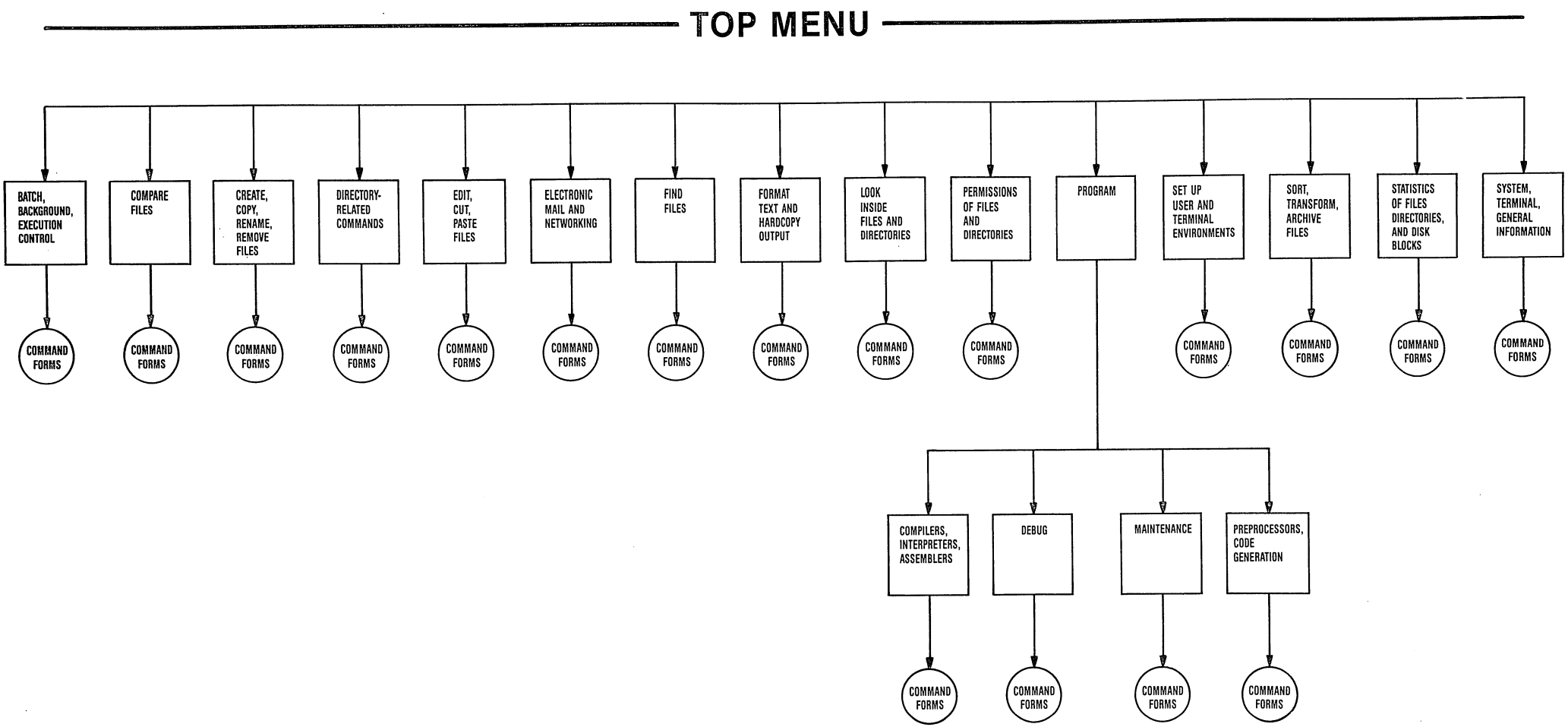


Figure 7-6: Command Form With Item Help

How To Exit Item/Field Help

Erase the current field help in the same way you erase the ASSIST commands help and the menu/form help. Strike **f2**, **^V**, or type any ASSIST command.

APPENDIX A: MENU HIERARCHY



NAME

assist — assistance using UNIX system commands

SYNOPSIS

assist [-s | [-c] [*name*]]

DESCRIPTION

The **assist** command invokes the ASSIST menu interface software for the UNIX system. The ASSIST menus categorize UNIX system commands according to function in a hierarchy. The menus lead to full-screen forms that aid you in the execution of a syntactically-correct UNIX system command line.

If you type **assist** without options, you enter at the top of the menu interface hierarchy. New users may select an introductory tutorial explaining how to use the ASSIST software. There are two options for ASSIST:

-c *name*

Use **-c** to invoke a command form in the user's current directory. *name* is an ASSIST-supported UNIX system command or the name of a walkthru.

-s Use **-s** to reinvoke the ASSIST setup module and check/modify your terminal variable. You can also access the introductory information about ASSIST by using **-s**.

When you invoke **assist**, you perform operations within the program by using **assist** commands. To see a list of the **assist** commands, strike **^A** or **f8** when you are in **assist**. When you do this, a list of the commands is printed on the terminal screen. The entire set of commands is described in the Glossary of ASSIST Commands in the *ASSIST Software Release 1.0 User's Guide*.

EXAMPLE

This example illustrates how to go directly to a particular command form. In this case, **mkdir** is the desired command form.

assist mkdir

SEE ALSO

astgen - ASSIST menu/form generator program

Appendix C: Setting Up

To ensure that ASSIST will work effectively with your terminal, ASSIST does some preliminary checks and verifications with a module in ASSIST called setup. Setup checks to see what type of terminal you are using and what special capabilities your terminal has, e.g., highlighting, function keys, and alternate character set. Setup does these checks by showing you a series of test patterns and asking questions about what you see on your screen. When you go through the setup initialization, the screens are self-explanatory. However, the screens that check the terminal, highlighting, and character set are shown in this guide because it is important for you to know the correct patterns.

Your UNIX system has a directory system that contains **terminfo** files. These files contain information about terminals. Each file defines such things as the number of columns on the screen, characters strings needed to perform functions such as clearing the screen, etc. The system you are using may not have information defining your terminal's capabilities. In this case, setup exits and a message appears on your screen which tells you that the ASSIST software will not work properly without terminal information on your system. If there is no **terminfo** file defining your terminal, contact your system administrator.

Once setup determines the characteristics of your terminal, the information is stored in a file called **.assistrc**

If you are using ASSIST for the first time, setup begins its initialization procedures at this point. Figures C-1 and C-2 show the screens for terminals that have been identified correctly and incorrectly, respectively.

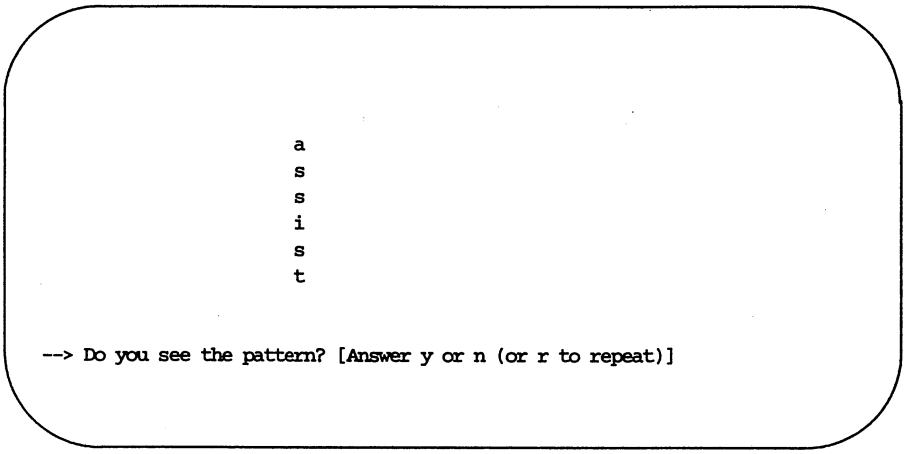


Figure C-1: Test Pattern for Correctly Identified Terminal

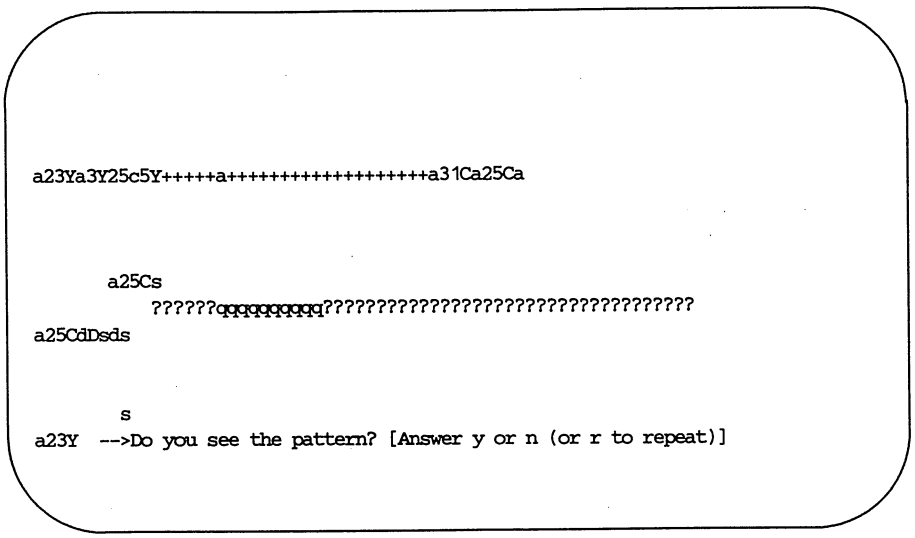


Figure C-2: Example of Pattern for Incorrectly Identified Terminal

Figure C-3 is an example of a highlighted screen.

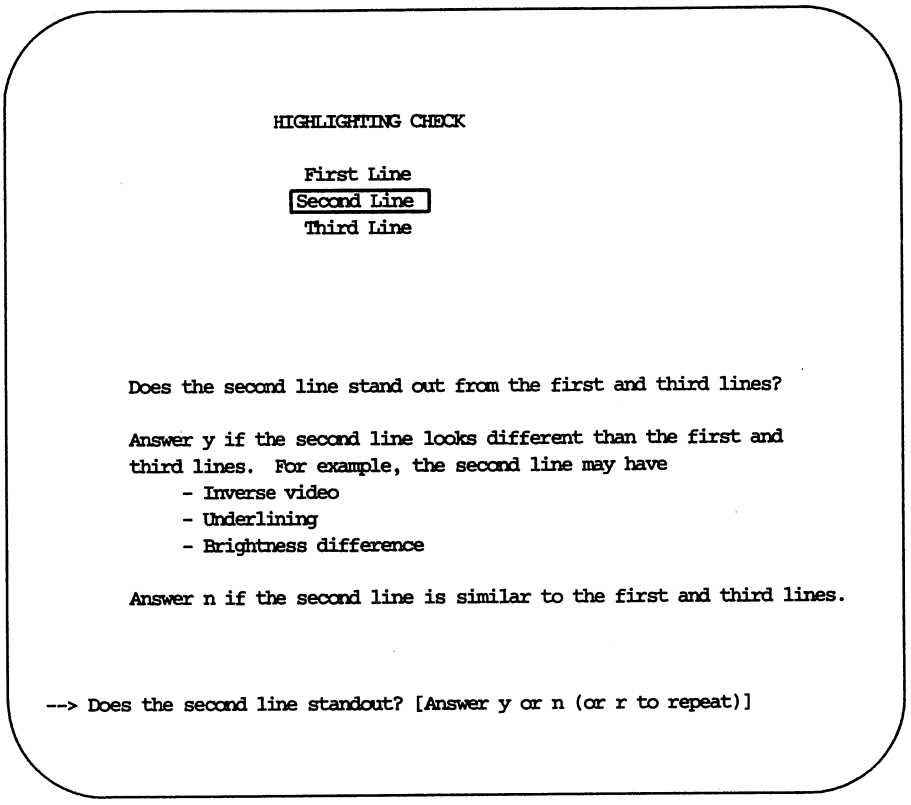


Figure C-3: Highlighting Test Pattern

Figure C-4 is an example of how the output looks on a terminal that has an alternative graphical character set.

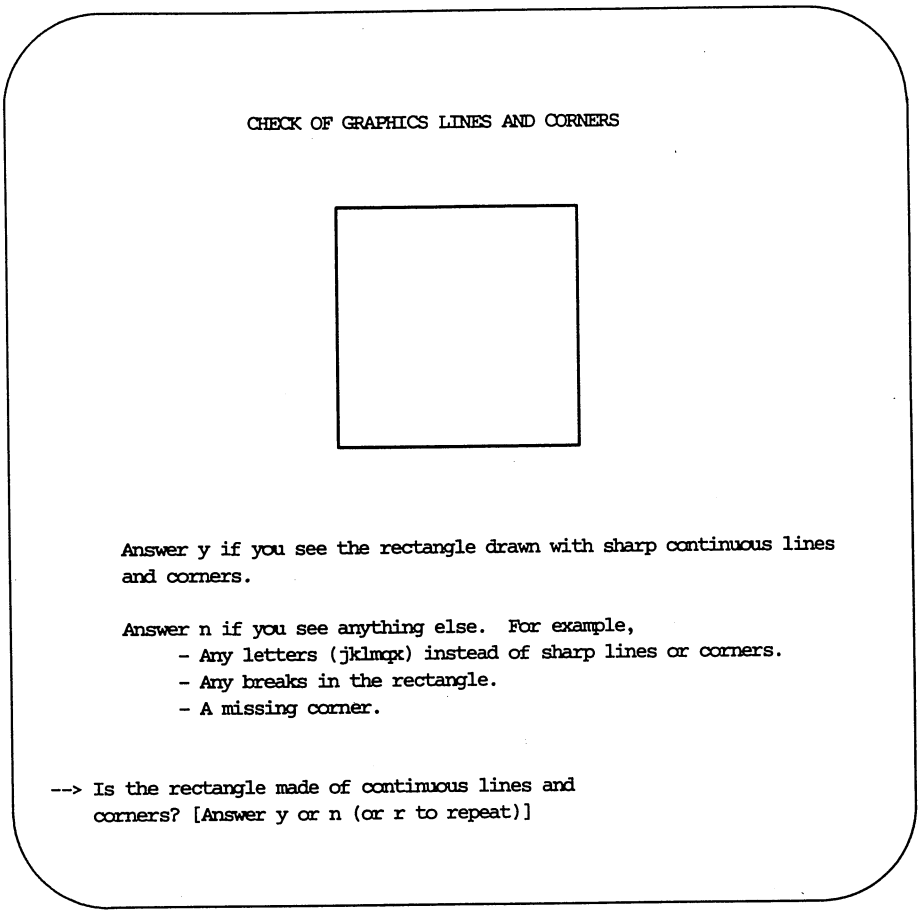


Figure C-4: Test Pattern For Alternative Character Set

Figure C-5 is an example of how the graphical character check looks on a terminal that does not have an alternative graphical character set.

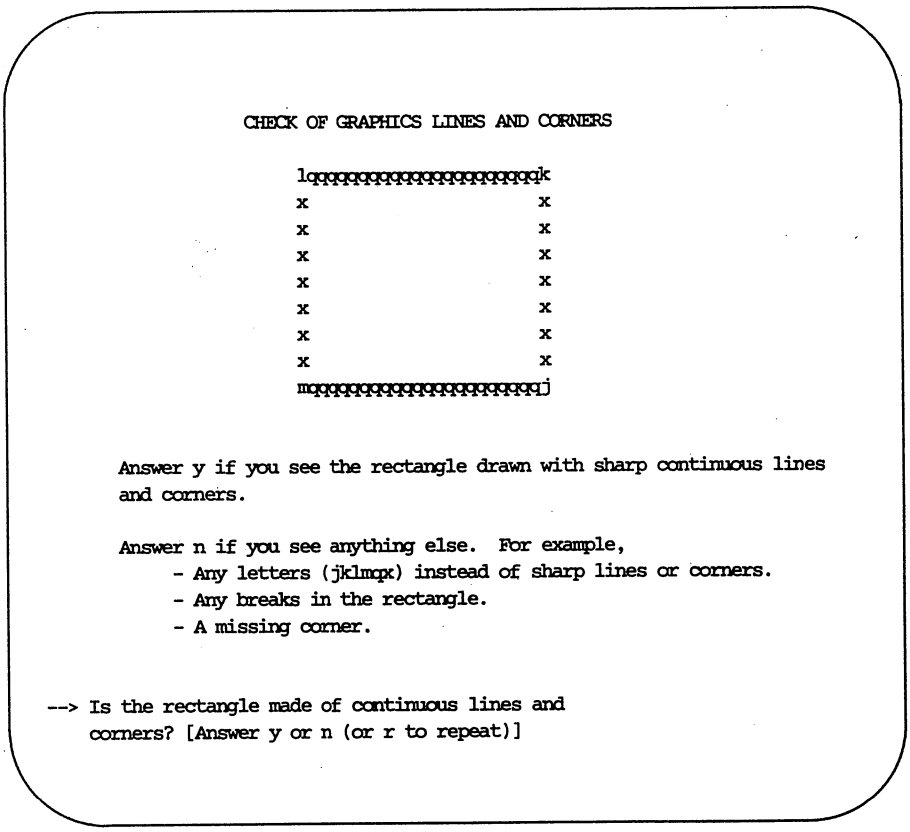


Figure C-5: Test Pattern For Non-alternative Character Set

After setup has checked to see if your terminal has an alternative character set, it checks to see if your terminal has function keys. If you strike the **f8** function key, as requested, and nothing happens, you have to strike another key. Setup will not do anything until you strike another key. If you strike **f8** and get the message

```
Sorry; got something other than f8
```

this means that the system did not understand the signals from your terminal's function keys.

If you do not have working function keys, this does not mean that you cannot use ASSIST. You can accomplish the same actions by using control characters instead of function keys. In fact, many people prefer to use the control characters. When you use control characters, you don't have to remove your hands from the keyboard.

One of the reasons that ASSIST does not recognize the signals your terminal is sending might be that the programmable function keys are set to something other than the **terminfo** default values. (To determine if your terminal has programmable function keys, refer to the user's manual or ask somebody who is familiar with the terminal.)

If you have programmable function keys that are programmed to something other than the **terminfo** default definition, ASSIST won't recognize the keys as working properly. For example, the **terminfo** definition of your f1 function key might be f1=ESC[A. If you defined the f1 function key as **f1=pwd; ls -l**, then ASSIST won't recognize your definition because it is not ESC[A.

On UNIX System V Release 3.0, there is a way around this. If you have a **.profile** file, add the following line

tput init

If you do not have a **.profile** file, type

tput init

after you log onto the computer. You must type this each time you log on if you have no **.profile** file.

tput init sets up your terminal so that programs like ASSIST can recognize your keys. However, on some terminals, **tput init** may erase what you have programmed into the function keys.

If you do not have **tput** on your system, you will get the following error message when you try to invoke it

```
tput: not found
```

If you get the message

```
tput: unknown capname init
```

this means that your system has an earlier version of **tput** than Version 5.3

and you cannot use it. If you are using Version 5.2, read the user's manual or ask somebody who is familiar with the terminal how to reset the function keys.

You may want to change some of the variables that describe your terminal to ASSIST. These variables are defined by ASSIST's setup module. You can directly access setup by typing

assist -s

You can perform three functions while in setup:

- Redefine your terminal type
- Identify your terminal's name
- Get introductory information about ASSIST

Appendix D: UNIX System Commands Supported by Command Forms

ar ..... Create and maintain groups of files in one archive file
awk ..... Pattern scanning and processing language
bfs ..... Big file scanner
cal ..... Show the calendar for a specified year
cat ..... Concatenate and show files
cc ..... C compiler
cd ..... Change working directory
chmod ..... Change permissions of files and directories
chown ..... Change owner of files and directories
cmp ..... Show byte and line number of differences
comm ..... Select/reject common lines
cp ..... Copy files
cpio ..... Copy file archives in and out
cpp ..... C language preprocessor
crypt ..... Encode or decode a file
csplit ..... Separate a file into sections
cu ..... Dial up another UNIX system
cut ..... Cut out selected fields of each line of a file
date ..... Show and set the date
df ..... Report number of free disk blocks
diff ..... Show all lines that differ
dircmp ..... Compare directories
du ..... Summarize dis usage
echo ..... Write arguments to the standard output
ed ..... Line editor
egrep ..... Search files using a regular expression
env ..... Set environment for command execution
export ..... Make the values of variables available to all sub-shells
fgrep ..... Search files for lines matching a pattern
file ..... Display file type
find ..... Find files
grep ..... Search files for lines matching a pattern
id ..... Print user and group IDs names
kill ..... Terminate a process
ld ..... Link editor for common object files
lex ..... Generate code for a lexical analysis of text
lp ..... Send files to a line printer

ln ..... Link files
lpstat ..... Report line printer status informaiton
ls ..... List contents of directory
mail ..... Send mail to user or read mail
make ..... Maintain, update, and regenerate groups of programs
mesg ..... Permit or deny messages
mkdir ..... Make a directory
mm ..... Format text with mm macros using nroff and troff
mv ..... Rename files
nohup ..... Run a command immune to hangups and quits
nroff ..... Format text containing nroff commands
od ..... Octal dump
passwd ..... Change login password
paste ..... Concatenate files into columns
pg ..... Examine files on screen-full at a time
pr ..... Print files on standard output
ps ..... Report process status
pwd ..... Working directory name
rm ..... Remove files
rmdir ..... Remove directories
sdb ..... Symbolic debugger for C and Fortran
sed ..... Stream editor
sh ..... Execute commands read from a terminal or file
size ..... Print section sizes of common object files
sort ..... Sort and/or merge the lines of a file
spell ..... Find spelling errors
split ..... Split a file into two pieces
strip ..... Strip symbol and line numbers from object files
stty ..... Set options for a terminal
su ..... Become super user or anther user
tail ..... Show last part of a file
umask ..... Set file creation permissions
uname ..... Show name of current UNIX system
uucp ..... UNIX system to UNIX system file copy
vi ..... Screen editor
wc ..... Report number of characters, words, and lines in a file
who ..... Show who is on the system
yacc ..... Yet another compiler compiler

Appendix E: ASSIST Error Messages

- ASSIST CANNOT FIND TERMINFO INFORMATION FILE FOR <*name*> TERMINAL TYPE

  Explanation: Your system's **terminfo** file does not contain information about the type of terminal you are using.

  Corrective Actions:
  Use **assist -s** to identify known **terminfo** entries.

- ASSIST USAGE:

```
assist ....................Enter ASSIST at top-level menu
assist [command_name] .....Enter command-form for command_name
assist -c [command_name] ....Use current directory for data file
assist -s ..................Invoke ASSIST's setup activities (terminal
                                    checking, introductory information)
```

  Explanation: You did not use the proper syntax when you typed the **assist** command.

  Corrective Actions:
  Use one of the command lines listed above.

- AT LEAST ONE OF THE HIGHLIGHTED ITEMS IS REQUIRED

  Explanation: You have not supplied input in a field where it is required. You must supply input for one of the highlighted fields before executing the command form.

  Corrective Actions:
  Enter input in at least one of the highlighted fields.

- AT LEAST ONE OF THE HIGHLIGHTED ITEMS ON THIS AND ON THE NEXT PAGE IS REQUIRED

  Explanation: An input entry is required in one of the fields highlighted on this page or on the next page.

  Corrective Actions:
  Enter input in at least one of the highlighted fields. on the current page or on the next page.

- BACKSPACE IS USED ONLY IN COMMAND FORMS (WHILE ENTERING AN INPUT STRING)

  Explanation: You are using **<BACKSPACE>** while you are on a menu. You can only use **<BACKSPACE>** in a command form.

  Corrective Actions:
  Use either of the following keys to move the cursor:
  RETURN, ^N, DOWN-ARROW, ^P, UP-ARROW

- BACKSPACE IS USED ONLY WHEN ENTERING INPUT STRING

  Explanation: When you are in a command form, you can only use **<BACKSPACE>** when you are entering input in a field. You cannot use **<BACKSPACE>** to control cursor movement.

  Corrective Actions:
  Use either of the following keys to move the cursor:
  ^B, ^N, ^P, ^W, RETURN, UP-ARROW, DOWN-ARROW

- ^C IS USED ONLY IN COMMAND FORMS (TO QUOTE CHARACTERS THAT ARE ASSIST COMMANDS

  Explanation: You are in a menu and using **^C**. You can only use **^C** in command forms.

  Corrective Actions:
  Use another ASSIST control key that is valid on menus.

- CANNOT MOVE FARTHER TO LEFT

  Explanation: The cursor is positioned as far to the left as possible. You are trying to backspace beyond the possible limits.

  Corrective Actions:
  Do not insert text at the current cursor position.

- CANNOT MOVE FARTHER TO RIGHT

  Explanation: The cursor is positioned as far to the right as possible. You are trying to space beyond the possible limits.

  Corrective Actions:
  Do not insert text at the current cursor position.

- *x* CANNOT OCCUR BOTH IN CURRENT AND PREVIOUS ITEM

  Explanation: This particular field does not allow a value identical to the previous field.

  Corrective Actions:
  Type in a new string that does not match any previous strings.

- CONTAINS A *x*, MUST BE ENCLOSED BY *y*

  Explanation: You entered a special character that requires shell quoting.

  Corrective Actions:
  Quote the special character.
  Use a different character.

- DIRECTORY DOES NOT EXIST OR IS NOT ACCESSIBLE, TYPE <CR>

  Explanation: You are trying to move to a directory that does not exist or is not accessible because of access permissions.

  Corrective Actions:
  Change access permissions for the directory you want to access; Use the UNIX system command **chmod**
  Fix typing errors.
  Enter a different directory name.

- DIRECTORY *x* HAS NO WRITE PERMISSION

  Explanation: You are trying to write in a directory that has no write permission.

  Corrective Actions:
  Change directory permissions; Use the UNIX system command **chmod**.
  Enter a different directory name.

- *x* DOES NOT EXIST

  Explanation: The file or directory name printed in the message does not exist.

  Corrective Actions:
  Fix typing errors.
  Enter another file/directory name.

- *x* DOES NOT HAVE EXECUTE OR SEARCH PERMISSION

  Explanation: The file/directory you have named does not have execute permission specified as one of its access permissions. You cannot execute this file/directory unless the permissions are changed to include execute permission.

  Corrective Actions:
  Change access permissions — use the UNIX system command **chmod**.
  Enter another file/directory name.

- DOES NOT HAVE READ PERMISSION

  Explanation: The file/directory you have named does not have read permission specified as one of its access permissions. You cannot read this file/directory unless the permissions are changed to include read permission.

  Corrective Actions:
  Change access permissions — use the UNIX system command **chmod**.
  Enter another file/directory name.

- *x* DOES NOT HAVE WRITE PERMISSION

  Explanation: The file/directory you have named does not have write permission. You cannot write in this file/directory unless the permissions are changed to include write permission.

  Corrective Actions:
  Change access permissions - use the UNIX system command **chmod**.
  Enter another file/directory name.

- *x* EXISTS

  Explanation: You have entered the name of a file.directory that already exists. You may already know that it exists. In case you are unaware of the file's/directory's existence, the message acts as a reminder. You may still want to overwrite or you may want to change your mind before the file is destroyed

  Corrective Actions:
  Change directories.
  Enter another file/directory name.
  Ignore the message if it does not matter that the file/directory already exists.

- FILE *x* HAS NO WRITE PERMISSION

  Explanation: You have entered the name of a file that does not allow you to write in it. You cannot write in this file unless the permissions are changed to include write permission.

  Corrective Actions:
  Change file permissions - use the UNIX system command **chmod**.

- FIRST HAVE WORD VALIDATED BY STRIKING <CR>

  Explanation: You must validate the last piece of date entered before you backup. You can backup to insert more data after the validation is done.

  Corrective Actions:
  Strike **<CR>**.

- *x* HAS EXECUTE OR SEARCH PERMISSION

  Explanation: This message is a verification that the file/directory has execute or search permission.

  Corrective Actions:
  None needed

- *x* HAS MORE THAN *n* CHARACTERS

  Explanation: The string you entered has too many characters.

  Corrective Actions:
  Enter a shorter string.

- *x* HAS READ PERMISSION

  Explanation: This message is a verification that the file/directory has read permission.

  Corrective Actions:
  None needed

- *x* HAS WRITE PERMISSION

  Explanation: This message is a verification that the file/directory has read permission.

  Corrective Actions:
  None needed

- INCOMPATIBLE WITH HIGHLIGHTED ITEM(S)

  Explanation: You have chosen an option whose function is incompatible with that of the highlighted option(s).

  Corrective Actions:
  Delete one of the incompatible options.

- INCOMPATIBLE WITH ITEM(S) ON OTHER PAGE(S)

  Explanation: You have chosen an option whose function is incompatible with that of options on another page.

  Corrective Actions:
  Reselect options.

- INPUT *x* AND TARGET *y* ARE THE SAME

  Explanation: You have specified the same file/directory as both the input and output. The input file/directory cannot also be the output file/directory.

  Corrective Actions:
  Enter non-identical input and output file/directory names.

- INPUT REQUIRED HERE

  Explanation: Some form of input is required for this field. You must enter something here.

  Corrective Actions:
  Enter data for this field.

- INPUT SHOULD BE *a, b, c,* OR *z*
  Explanation: Only certain input values are acceptable. They are listed in the error message.

  Corrective Actions:
  Supply one of the acceptable values.

- INPUT SHOULD BEGIN WITH *string1, string2 ...* OR *stringn*
  Explanation: The input should begin with a specific string. The acceptable strings are listed in the error message.

  Corrective Actions:
  Prefix your input with one of the strings listed in the message.

- INPUT SHOULD NOT BE *a, b, c,* OR *z*
  Explanation: The input should not be certain values. They are listed in the error message.

  Corrective Actions:
  Input any value except one of the ones listed in the error message.

- INPUT SHOULD NOT BEGIN WITH *string1, string2, ...* OR *stringn*
  Explanation: The input should not begin with certain strings. They are are listed in the errror message.

  Corrective Actions:
  Do not prefix your input with any of the strings listed in the error message.

- INPUT SHOULD END WITH *string1, string2, ...* OR *stringn*
  Explanation: The input should end with one of a set of specified strings. Your input does not end with one of the strings listed in the message.

  Corrective Actions:
  Suffix your input with one of the strings listed in the error message.

- INPUT SHOULD NOT END WITH *string1, string2, ...* OR *stringn*
  Explanation: The input should not end with one of a set of listed strings. The strings are listed in the error message.

  Corrective Actions:
  Do not end your input with any of the strings listed in the error message.

- INPUT STRING MUCH TOO LONG
  Explanation: There is a maximum length for the string. You have exceeded the maximum length.

  Corrective Actions:
  Enter a shorter string.

- *x* IS A DIRECTORY
  Explanation: You have specified a directory. A directory name is not the expected input.

  Corrective Actions:
  Enter a file name.

- *x* IS NOT A DIRECTORY
  Explanation: You have entered a name which is not a directory name.

  Corrective Actions:
  Enter a directory name.

- *x* IS NOT A VALID ASSIST COMMAND
  Explanation: The only valid characters for a *first-letter* search are the letters a-z and A-Z. You enterd a character that was not one of the valid ones.

  Corrective Action:
  Type the first letter of the item that you wish to choose.

- ^K IS USED ONLY IN COMMAND FORMS (TO STORE COMMAND LINE IN A FILE)

  Explanation: You used ^K on a menu. ^K is only valid on command forms.

  Corrective Actions:
  Strike the appropriate ASSIST command key that applies to menus.

- NO ITEM ON THIS MENU STARTS WITH *x*

  Explanation: In a *first-letter* search, you typed a letter that does not have corresponding menu item.

  Corrective Actions:
  Type the first letter of the item that you wish to choose.

- NOT A VALID COMMAND ON POPUP MENU

  Explanation: You used a command that is invalid on the pop-up menu.

  Corrective Actions:
  Choose a valid pop-up menu command.

- NOT ACCEPTABLE; CHECK ITEM HELP

  Explanation: You have entered an invalid string. The list of acceptable strings or formats is too long to list in the the error message. You must refer to the item help for more information.

  Corrective Actions:
  Enter one of the acceptable strings listed in the item help.
  Menu/form item help - strike ^Y or f6

- NOT USED IN MENUS

  Explanation: You have typed a command that is not used on menus.

  Corrective Actions:
  Enter a valid menu command.

- NOTHING TO ERASE

  Explanation: In a command form, <SPACE> erases the string on which the cursor is positioned. You struck <SPACE> when when the cursor was resting on a null string. Therefore, there is nothing to erase.

  Corrective Actions:
  Move to another field.
  Type in a string.

- ONE STRING ONLY

  Explanation: ASSIST expects only one string in this field. You tried to enter more than one string.

  Corrective Actions:
  Enter only one string. If your string contains a space, surround the string with quotes.

- REQUIRED BY HIGHLIGHTED ITEM(S)

  Explanation: Data is required in this field by the highlighted item(s) on the command form. You must enter data in this field.

  Corrective Actions:
  Enter data in the current field.
  Select different options.

- REQUIRED BY HIGHLIGHTED ITEM(S) ON OTHER PAGE

  Explanation: Data is required in this field by the highlighted item(s) on another page on the command form. You must enter data in this field.

  Corrective Actions:
  Enter input in the current field.
  Select different options.

- REQUIRES HIGHLIGHTED ITEM(S)

  Explanation: This field requires input entries in other fields. The fields where an entry is required are highlighted. You must enter input in these fields.

  Corrective Actions:
  Enter data in the highlighted fields.
  Select different options.

- REQUIRES ITEM(S) ON OTHER PAGES

  Explanation: This field requires input entries on other pages of the command form. You must enter input in these fields.

  Corrective Actions:
  Enter input in the required fields.
  Select different options.

- *x* SHOULD BE *y*

  Explanation: You typed *x*. You need to type *y* instead.

  Corrective Actions:
  Type *y*

- SHOULD BE *x*-SEPARATED LIST OF: *y,z,...*

  Explanation: The items on the list you entered are separated by the wrong character or one or more members of the list are invalid. For example, you may have entered a list whose items were separated by spaces when commas were required.

  Corrective Actions:
  Use the character shown in the error message.
  Re-enter the list.

- <SPACE> IS USED ONLY IN COMMAND FORMS

  Explanation: You struck **<SPACE>** when you were not in a command form. **<SPACE>** is only valid in command forms.

  Corrective Actions:
  Strike the appropriate ASSIST command.

- THERE IS ONLY ONE PAGE

  Explanation: You have tried to go to the next page when there is only one page.

  Corrective Actions:
  Strike the appropriate command to perform the next operation you want to do.

- TYPE NAME OF EXISTING DIRECTORY

  Explanation: ASSIST expects the name of an existing directory. You entered a non-existing name.

  Corrective Actions:
  Correct typing errors.
  Enter name of an existing directory.

- VALIDATION INTERRUPTED

  Explanation: While ASSIST was validating your input in a command form, there was an interruption. You may have accidentally hit the DEL or BREAK key.

  Corrective Actions:
  None required, continue what you were doing.

- WOULD OVERWRITE *x*

  Explanation: You have typed the name of a file/directory that already exists. The existing contents of this file/directory will be destroyed.

  Corrective Actions:
  Enter a new file/directory name.

- YOU DO NOT OWN *x*

  Explanation: You are trying to perform an operation on a file/directory that you do not own. Only the owner may operate on the file/directory.

  Corrective Actions:
  Enter a different file/directory name.

Appendix F: Summary of Commands Used in Walkthrus

Summary of ASSIST Commands Described in Introduction To ASSIST Walkthru

f1, ^G ..... Select item (menus) or execute command (command forms)
f4, ^R ..... Previous menu/form
f5, ^F ..... Invoke pop-up menu
f6, ^Y ..... Current item help
f7, ^O ..... Current menu/form help
f8, ^A ..... ASSIST command help
RETURN ..... Next item

Summary of UNIX System Commands Described in UNIX System Walkthru

ls ..... List the names of files in a directory
cat ..... Display the contents of a file
pwd ..... Print the name of the current directory
cd ..... Change directory
mkdir ..... Make a new directory
> ..... Erase old contents of *filename* and save output of *command* in *filename*
>> ..... Append output of *command* to current contents of *filename*
< ..... Redirect input from *filename* rather than standard input
| ..... Pipe output from *command1* instead of standard input to *command2*

Summary of Commands Described in vi Walkthru

Cursor Moving Commands

SPACE ..... Forward one character
<BS> ..... Backward one character
j ..... Jump-down one line
k ..... Kick-up one line
w ..... Word forward
b ..... Backward-word
6k ..... Move down 6 lines
4b ..... Move back 4 words

Deleting Commands

x ..... x-out character
dw ..... Delete word
dd ..... Delete entire line
3x ..... Delete (x-out) 3 characters
6dw ..... Delete 6 words
4dd ..... Delete 4 lines

Adding Text Commands

i ..... Insert text to left of cursor
a ..... Add text to right of cursor
o ..... Open line below current line
O ..... Open line above current line
<BS> ..... Erase characters in input mode
<ESC> ..... Stop inputting text; return to command level of **vi**

Writing and Quitting

:w<CR> ..... Save (write) changes
:q<CR> ..... Quit **vi**
:q!<CR> ..... Quit **vi** without saving the changes made since the last write command

Undo Command

u ..... Undo last editing change
u ..... Redo (undo last undo)

Options

:set nu<CR> ..... Turn on line numbering
:set nonu<CR> ..... Turn off line numbering
:set smd<CR> ..... Turn on show mode
:set nosmd<CR> ... Turn off show mode

Summary of Commands Described in sdb Walkthru

Printing Commands

p ..... Print the current line
Np ..... Make N the current line and print it
w ..... Print a window of 10 lines around the current line
^D ..... Scroll 10 lines and advance the current line by 10
/PATTERN ..... Search forward for the next line containing PATTERN
?PATTERN ..... Search backward for the next line containing PATTERN

Breakpoint Debugging Commands

b ..... Set breakpoint at current line
Nb ..... Set breakpoint at line N
B ..... List all breakpoints
d ..... Delete all breakpoints

Executing and Finding Current Variable Value Commands

r ..... Execute command
c ..... Continue executing stopped command
s ..... Execute a single step of a command
VARNAME/ ..... Find current value of variable VARNAME

Debugging Code with More Than One Source File

e *filename* ..... Make *filename* the current file
e ..... Print name of current function and file

Debugging Core Dumps

t ..... List function calls leading to error that caused dump

ASSIST Function Keys

f1 Function key **f1** selects the current item on a menu. On a command form, **f1** executes a command line. **f1** has the same functions as the ^G ASSIST command.

f2 Function key **f2** clears a pop-up help message, clears the pop-up menu, or kills a prompt line. **f2** has the same function as the ^V ASSIST command.

f3 Function key **f3** redraws the screen. Pop-up help or messages are not redrawn. **f3** has the same function as the ^L ASSIST command.

f4 Function key **f4** returns to the previous menu or form screen. When you are in the TOP menu and strike **f4**, you receive a message telling you that you are about to exit ASSIST. **f4** has the same function as the ^T ASSIST command.

f5 Function key **f5** invokes the pop-up menu. **f5** has the same function as the ^F ASSIST command.

f6 Function key **f6** displays the pop-up help message for the current item on a menu or command form. The current item is the item on which the cursor is resting. **f6** has the same function as the ^Y ASSIST command.

f7 Function key **f7** displays the pop-up help message for the current menu or command form. **f7** has the same function as the ^O ASSIST command.

f8 Function key **f8** displays the pop-up help message list of ASSIST commands. **f8** has the same function as the ^A ASSIST command.

ASSIST Control-Character Commands

^A Control-a (Aid) displays the pop-up help message list of ASSIST commands. It has the same function as **f8**.

^B Control-b (Backword) moves the cursor back to the previous word on a command form.

^C Control-c (Character) is used in command forms to indicate characters that are ASSIST commands. For example, if you type **^B** on a command form, you must precede it with **^C**.

^D Control-d (Depart, compatible with the UNIX system) exits the ASSIST program and returns you to the shell.

^E Control-e (Escape) is a temporary shell escape. It allows you to escape ASSIST and execute one shell command. When you strike **^E**, it produces the prompt `TYPE UNIX COMMAND:`. The **^E** prompt accepts any shell command line or **^V** or **f3** to kill the prompt. After you execute the single shell command, you return to ASSIST.

^F Control-f (Functions) invokes the pop-up menu. It has the same function as **f5**.

^G Control-g (Go) selects the current item on a menu. It also executes a command line on a command form. It has the same function as **f1**.

^K Control-k (Keep) stores a command line that you have built using a command form. It stores it in a temporary file called *commandname*.**assist**.

^L Control-l (Look-again, compatible with the UNIX system) redraws the screen. Pop-up help or messages are not redrawn. It has the same function as **f3**.

^N Control-n (**N**ext) moves the cursor to the next item or field below the current item. If the current item or field is the last item on a menu or command form, the cursor moves to the first or next page.

^O Control-o (**O**verview) displays the help message for the current menu or command form. It has the same function as **f6**.

^P Control-p (**P**revious) moves the cursor to the item just above the current item on a menu or command form. If the current item is the first item, the cursor moves to the last item.

^R Control-r (**R**eturn) returns to the previous menu or form. If there is not a previous menu or screen, e.g., the current screen is the point at which you entered ASSIST, **^R** gives you a prompt and asks if you want to exit. It has the same function as **f4**.

^T Control-t (**T**OP) returns to the TOP menu screen.

^U Control-u (**U**p) moves to the next page of the current menu or command form.

^V Control-v (**V**anish) clears a pop-up help message, the pop-up menu, or a prompt line. It has the same function **f2**.

^W Control-w (**W**ord) moves to the next word on a command form.

^Y Control-y (**Why**) displays the item pop-up help message for the current item on a menu or command form.

Other Commands

<RETURN>	Carriage return moves the cursor down to the next item or field.
<SPACE>	The **<SPACE>** key is used on command forms to edit a field and indicate the termination of a word. If a field already has an entry **<SPACE>** will erase the entry.
<DOWN-ARROW>	The **<DOWN-ARROW>** key moves the cursor to the next input field on a command form or the next item on a menu.
<UP-ARROW>	The **<UP-ARROW>** key moves the cursor to the input field or item just above the current one.
<LEFT-ARROW>	The **<LEFT-ARROW>** key moves the cursor to the left on command forms. It is not used on menus.
<RIGHT-ARROW>	The **<RIGHT-ARROW>** key moves the cursor to the right on command forms. It is not used on menus.

Glossary of Terms

alternate character set
The alternate graphical character set gives you enhanced graphics on your terminal.

command form
An ASSIST command form is a full-screen form that helps you build and execute a UNIX system command line.

command line
When you want to execute a UNIX system command, you must supply the command name and any options. This is done on the command line. When you type the command name and options on the command line and strike **RETURN**, the UNIX system command executes.

command search
Command search is a feature on the ASSIST pop-up menu that helps you find the name of a UNIX system command that performs a particular function.

context-sensitive
The information that you receive is dependent on where you are in ASSIST. ASSIST automatically knows if you are in a menu or command form and gives you information that is appropriate for your location.

control character
A control character is entered when you depress the **CTRL** key and strike an unshifted letter simultaneously. Control characters in ASSIST serve as commands. ASSIST uses control characters to call for the various forms of pop-up help and for navigating on the screen. A control character in this document is represented by a caret (^) and an upper-case letter, e.g., ^D.

current item
The current item in ASSIST is the item on which the cursor is resting. In ASSIST, it is represented by the greater-than sign (>).

cursor
The cursor indicates where you are on the screen. On a menu, the cursor rests in front of the item. On a command form, the cursor rests at the first position of an input field.

exit message — An ASSIST exit message tells you how to exit from an interactive command before you execute it. It only appears when you need to do more than strike **RETURN**. For example, whenever the command expects input from the standard input, you are put into an interactive mode. You cannot exit this mode by striking **RETURN**. You must strike **^D**. An exit message gives you this information.

field — ASSIST command forms contain descriptive lines of text called fields. Fields describe command options and prompt you for a response in selecting the options. Your response may be in the form of a yes/no answer or you may enter text.

full-screen menu — ASSIST full-screen menus occupy the entire area of your terminal's screen. These menus contain selectable items and are the mechanism for navigating through ASSIST.

function keys — Function keys are usually located on the top row of your terminal's keyboard. They may be labelled f1 through f8. ASSIST assumes that your terminal has eight function keys. If ASSIST knows that your terminal has eight working function keys, then ASSIST displays f8 as well as **^A** in the upper right corner of the menus and command forms. Each key performs a specific action. See the Glosary of ASSIST Commands for a description of their functions.

highlighting — Some terminals have the capability to make designated pieces of text standout from the rest of the text on the screen. If, for example, the screen background is black and the characters are white, when a particular area is highlighted, the background for that area will be white and the characters black.

insert mode — When you are in the insert mode, every character you type is entered as a piece of text. Text editors such as **ed** and **vi** use the insert mode when you type your text into a file.

item ASSIST Menus contain a list of choices from which you make a selection. These choices are called items on the menu. Each item can be selected.

modem A modem is a piece of equipment that transmits data from your terminal to a computer. It does this by sending your data over the telephone lines.

pipe A UNIX system pipe connects the output of one UNIX system command to the input of another UNIX system command without using a temporary file.

pop-up help Help is available for ASSIST commands, menus/forms, and items. When you ask for help, a box containing the information "pops up" onto the screen.

pop-up menu The pop-up menu is not a full-screen menu. When you access the menu, a small box containing the menu "pops up" onto the screen. The cursor is placed on the pop-up menu. The pop-up menu provides direct access to many of ASSIST's functions.

printable character Not every key or character you strike causes something to be printed on the screen. For example, when you strike **TAB**, **RETURN**, or **BACKSPACE**, nothing is printed on your screen. A printable character is one that appears on the screen when you strike it on the keyboard.

redirection Most UNIX system commands take their input from the terminal and produce output on a terminal. The terminal can be replaced by a file for either or both input and output. This is redirection. Input redirection is specified by the less-than sign (<) and a filename. Output redirection is specified by the greater-than sign (>) and a filename or (>>) and a filename. The (>>) means that the output should be attached to the end of the file named.

searchword A searchword is a unique word in the description of a UNIX system command. The searchword should be as unique as possible to the description of the command. One of ASSIST's features, the command search, helps you identify the name of a command that performs a specific function. ASSIST searches a database containing the desriptions of UNIX system commands. It reports all commands that have the searchword or its synonym in their description. A searchword is also referred to as a keyword.

setup Setup is a module in ASSIST that checks to see what type of terminal you are using and what special capabilities your terminal has. It ensures that ASSIST will work effectively with your terminal.

shell The shell sits between you and the program that controls the resources of your computer and allocates them among its users. The shell is a program that acts as an interpreter. It also lets you do input/output redirection, define your own commands, and find filenames using pattern matching.

shell escape The shell escape takes you out of whatever you are currently doing and puts you temporarily at the shell level.

shell expansion Shell expansion expands the environmental variables such as $HOME. It also expands all special characters. This means, for example, that $HOME is translated into your full HOME directory pathname.

standard input Standard input is input that you provide to the computer. The input is assigned a file descriptor, 0, as a means of identification. When you type input into the computer directly from your terminal keyboard, file descriptor 0 is used by default. You may redirect the standard input so that it is read in from a file instead of the terminal. In this case, the file descriptor is the name you have given the input file.

standard output Standard output is output that you receive from the computer. The output is assigned a file descriptor, 1, as a means of identification. When you receive output from the computer that is printed on your terminal, it is assigned file descriptor 1 by default. You may redirect the output into a file instead of receiving it at the terminal. In this case, the file descriptor is the name you have given the output file.

validation When you are using a command form to build and execute a command line, ASSIST checks each piece of input that you enter. If the entry is incorrect, ASSIST gives you a message telling you about the problem. This checking process is call validation.

video display terminal (VDT)
A video display terminal is a terminal with a screen.

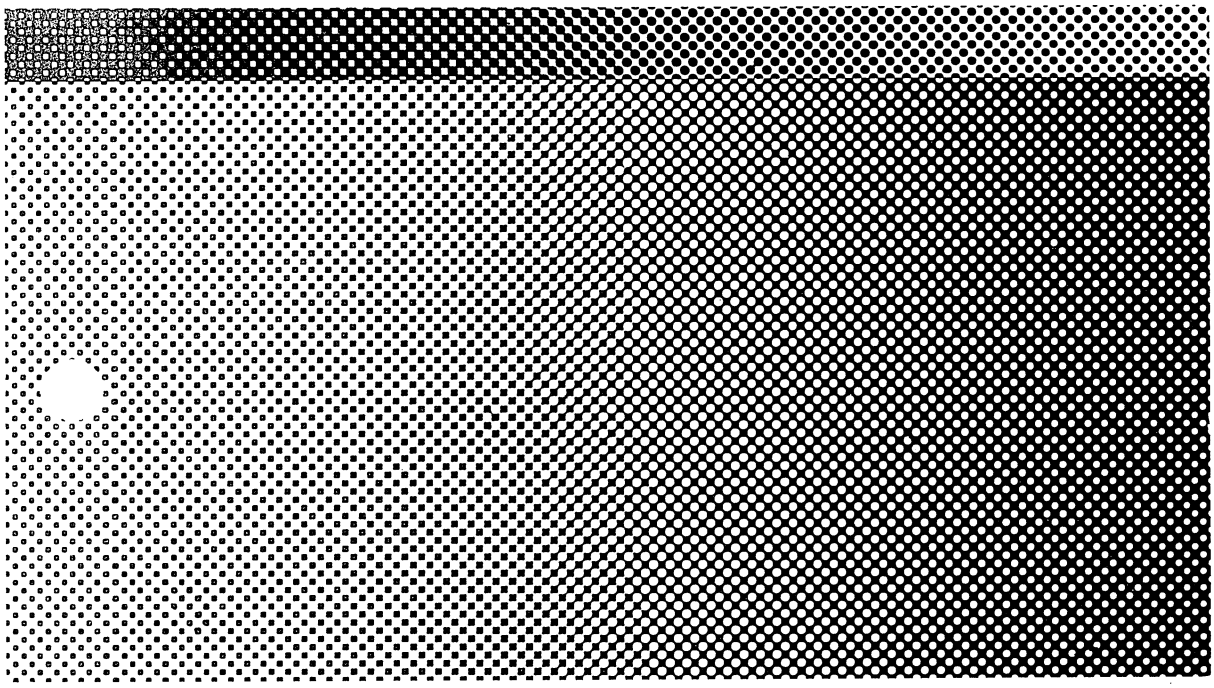
walkthru An ASSIST walkthru is designed to give you introductory information about commands and concepts for a particular subject. It provides this information by "walking you through" a simulation of the command or concept.

Index

307-236

UNIX® System V
ASSIST Software Release 1.0

Source Code Release Notes



©1986 AT&T
All Rights Reserved
Printed in USA

NOTICE

The information in this document is subject to change without notice. AT&T assumes no responsibility for any errors that may appear in this document.

UNIX is a registered trademark of AT&T.

Table of Contents

Introduction

The ASSIST Software provides a menu/form interface to the UNIX system and a menu/form generator tool. The menu interface is designed to make the UNIX system easier to use by providing menus and command forms to assist you in locating a UNIX system command and building an executable UNIX system command line. The menu/form generator tool enables you to develop or modify the ASSIST menus and forms for your own systems. The ASSIST Release 1.0 source product is supported to build and run on the 3B2/300, 3B2/310, and the 3B2/400 Computers running UNIX System V Release 3.0. It is also supported to build on the 3B2/300, 3B2/310, and the 3B2/400 Computers running UNIX System V Release 2.0.4 and 2.1.

The ASSIST Software can be installed when you install the UNIX system. It can also be installed after you have installed the UNIX system or it can be installed by itself. The distinction between the last two cases is that in the first instance, you may have forgotten to install ASSIST at the time of the UNIX system installation. In this case, all of the libraries you need to install and build ASSIST are already there so you simply install ASSIST. In the second case, when you install ASSIST by itself on any 3B2 Computer, you also need to install the **libcurses** library. Although these two cases seem similar, they require different procedures to install and build.

Installation and Build Procedures

This section guides you through the installation procedure for ASSIST Software on the 3B2 Computer and in a non-3B2 Computer cross environment. The *UNIX System V Release 3.0 Source Code Provision Release Notes* provide instructions for installing and building the UNIX System. You will need to refer to those instructions along with the instructions provided here in the *ASSIST Release Notes* when installing the ASSIST Software. Before building the ASSIST Software, ensure that the following software utilities are installed on your system.

- Terminal Information Utilities
- Directory and File Management Utilities
- Essential Utilities
- C Programming Language Utilities Issue 4
- Advanced Programming Utilities Issue 1:
  - Extended Software Generation Utilities
- User Environment Utilities

Conventions Used in These Release Notes

Throughout the following procedures the information you type is in **bold** letters. The symbol **<CR>** represents the carriage return key. Be sure to type in the spaces as shown.

To differentiate between references to sections of the *UNIX System V Release 3.0 Source Code Provision Release Notes* and sections of this document, you will be pointed to "Sections" in the *Source Code Provision Release Notes* and "Sets" in this document.

When you install and build ASSIST, you should be logged in as **root** on the console terminal. However, on non-3B2 computers, it is not recommended that you log in as **root**.

Space Requirements

The space requirements differ, depending on the type of ASSIST build you want to do. Choose the type of ASSIST build and follow the instructions for that build. You will be

1. Building ASSIST *with* a full system install and build of System V Release 3.0, or
2. Building ASSIST *after* a full system install and build of System V Release 3.0, or
3. Building ASSIST only

The following information is divided into instructions for calculating space requirements based on the type of build.

1. With a Full Build of System V Release 3.0

Step 1: You need 41000 blocks and 5000 inodes in **/usr** to install the Source Code Provision cartridge tape (from the table in Figure 1 in the *Source Code Provision Release Notes*).

Step 2: Look at the table in Figure 2 of the *Source Code Provision Release Notes*. From the column headed "Running System", select the item that identifies the system you will be running. To calculate the space needed to setup and build the type of system you have chosen, select the numbers from **/usr** and **/(root)**.

Step 3: You will need an additional 8000 blocks and an additional 550 inodes to install and build the ASSIST Software.

Step 4: Add the numbers obtained from Steps 1, 2, and 3.

EXAMPLE: Space Requirements for Install and Build of System V Release 3.0 in **chroot** with ASSIST

	/usr		/(root)	
	blocks	**inodes**	**blocks**	**inodes**
install	41000	5000	0	0
setup	25700	1550	3300	36
build	18600	1400	0	0
ASSIST	8000	550	0	0
Total	93300	8500	3300	36

2. After a Full Build of System V Release 3.0

You need 8000 blocks and 550 inodes. This build assumes that **libcurses.a** has already been built. (You will not require any space in **/(root)**.)

3. ASSIST Only

If you are running UNIX System V Release 3.0 on your 3B2 Computer, **libcurses.a** will already be available. You can check this by executing

```
cd /
ls –l /usr/lib/libcurses.a
```

You need 8000 blocks and 550 inodes in **/usr**. (You will not require any space in **/(root)**.)

If you are running Release 2.1 or 2.0.4, you will need to build **libcurses.a**. Therefore, you will need 28000 blocks and 2500 inodes in **/usr**. (You will not require any space in **/(root)**.)

Estimates of Build Times

The following list gives an estimate of the time needed to do a full UNIX System V Release 3.0 build, a **libcurses** build, and an ASSIST build.

- Full UNIX system build - 27 hours
- **libcurses** build - 2 to 4 hours
- ASSIST build - 1 to 2 hours

Installation Considerations

Before you install the ASSIST source code, you must determine the appropriate procedure for installing and building ASSIST.

- If you are installing and building on a 3B2 Computer, you must know the UNIX System V release currently running. The procedure you follow depends on the release. The options are UNIX System V release 2.0.4, UNIX System V Release 2.1, or UNIX System V Release 3.0.
- You must decide whether you are going to install and build in the standard **root** directory or in another directory.
- You must decide whether you are going to install and build ASSIST as part of a full UNIX system installation, after a full UNIX system has been installed, or as ASSIST alone.

When you have made these decisions, consult the table below to find out which of the procedures to follow. In each case, go to the set named in the table and follow only the procedure detailed in that set.

Type of Build	Environment	UNIX Release or Non-3B2 Host	GO TO
Full System	Standard	3.0	Set 1
Full System	**chroot**	3.0	Set 2
Full System	**chroot**	2.1, 2.0.4	Set 3
Full System	Cross Environment	non-3B2 Host	Set 4
After Full System	Standard	3.0	Set 5
After Full System	**chroot**	3.0	Set 6
After Full System	**chroot**	2.1, 2.0.4	Set 7
After Full System	Cross Environment	non-3B2 Host	Set 8
ASSIST Only	Standard	3.0	Set 9
ASSIST Only	**chroot**	3.0	Set 10
ASSIST Only	**chroot**	2.1, 2.0.4	Set 11

Set 1: Full Release 3.0 Build in the Standard Root Directory on a 3B2 Computer Running 3.0

Step 1: Follow instructions 1 through 5 in Section 1 of the *Source Code Provision Release Notes.*

Step 2: Place the ASSIST cartridge tape in the tape drive.

Step 3: To install the ASSIST source code, enter the following command.

ctccpio –idumvT /dev/rSA/ctape1

Step 4: Continue with Step 6 in the *Source Code Provision Release Notes.* ASSIST will be built with the rest of the UNIX system commands.

Set 2: Full Release 3.0 Build in chroot on a 3B2 Computer Running 3.0

Step 1: Follow instructions 1 through 11 in Section 3 of the *Source Code Provision Release Notes.*

Step 2: Place the ASSIST cartridge tape in the tape drive.

Step 3: To install the ASSIST source code, enter the following command.

ctccpio –idumvT /dev/rSA/ctape1

Step 4: Continue with Step 12 in the *Source Code Provision Release Notes.* ASSIST will be built with the rest of the UNIX system commands.

If you want to incorporate ASSIST into the standard location after the **chroot** build has completed, go to the instructions in this document "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Set 3: Full Release 3.0 Build in chroot on a 3B2 Computer Running 2.1 or 2.0.4

Step 1: Follow instructions 1 through 14 in Section 5 of the *Source Code Provision Release Notes.*

Step 2: Place the ASSIST cartridge tape in the tape drive.

Step 3: To install theASSIST source code, enter the following command.

ctccpio −idumvT /dev/rSA/ctape1

Step 4: Continue with Step 15 in the *Source Code Provision Release Notes.* ASSIST will be built with the rest of the UNIX system commands.

If you want to incorporate ASSIST into the standard location after the **chroot** build has completed, go to the instructions in this document "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Set 4: Full Release 3.0 Build in a Cross Environment on a non-3B2 Computer

Step 1: Follow instructions 1 through 8 in Section 7 of the *Source Code Provision Release Notes.*

Step 2: Load the 9-track magnetic tape on which the ASSIST source code was delivered onto drive 0 of the host machine.

Step 3: Install the ASSIST source code by executing

cpio −icdumvB < /dev/rmt/0m

Step 4: Continue with Step 9 in the *Source Code Provision Release Notes.* ASSIST will be built with the rest of the UNIX system commands.

Set 5: Build ASSIST After a Full Release 3.0 Build in a Standard Root Directory on a 3B2 Computer Running 3.0

This assumes that you have followed all of the instructions in Section 1 of the *Source Code Provision Release Notes.*

Step 1: Log in as **root.**

Step 2: Change directory to the **root** directory by executing

```
cd /
```

Step 3: Check to see if **libcurses.a** exists by executing

```
ls -l usr/lib/libcurses.a
```

Step 4: If **libcurses.a** exists, go to Step 5. If not, build the libraries by executing

```
cd usr/src
./:mklib libcurses
```

Step 5: Place the ASSIST cartridge tape in the tape drive.

Step 6: To install the ASSIST source code, enter the following commands.

```
cd /
ctccpio -idumvT /dev/rSA/ctape1
```

Step 7: Change to the directory where the ASSIST source code is located by executing

```
cd usr/src
```

Step 8: Build the ASSIST code by executing

```
./:mkcmd assist
```

Set 6: Build ASSIST After a Full Release 3.0 Build in chroot on a 3B2 Computer Running 3.0

Step 1: Follow instructions 1 through 15 in Section 3 of the *Source Code Provision Release Notes.*

Step 2: Change to **chroot** directory by executing

cd /

NOTE: If you have already exited the **chroot** environment with **control-d**, refer to Section 3 of the *Source Code Provision Release Notes* to reset environment variables and re-establish **chroot**.

Step 3: Place the ASSIST cartridge tape in the tape drive.

Step 4: To install the ASSIST source code, enter the following command.

ctccpio –idumvT /dev/rSA/ctape1

Step 5: Change directory to the directory containing the source code by executing

cd usr/src

Step 6: Build the ASSIST command by executing the following command.

./:mkcmd assist

If you want to incorporate ASSIST into the standard location after the **chroot** build has completed, go to the instructions in this document "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Set 7: Build ASSIST After a Full Release 3.0 Build in chroot on a 3B2 Computer Running 2.1 or 2.0.4

Step 1: Follow instructions 1 through 30 in Section 5 of the *Source Code Provision Release Notes.*

Step 2: Change directory to **$ROOT** by executing the following command.

cd $ROOT

NOTE: If you have already exited the **chroot** environment with **control-d**, refer to Section 5 of the *Source Code Provision Release Notes* to reset environment variables, e.g., **$ROOT**.

Step 3: Place the ASSIST cartridge tape in the tape drive.

Step 4: To install the ASSIST source code, enter the following command.

ctccpio –idumvT /dev/rSA/ctape1

Step 5: Change directory to the directory containing the source code by executing

cd $ROOT/usr/src

Step 6: Build the ASSIST command by execting the following command:

./:mkcmd assist

If you want to incorporate ASSIST into the standard location after the **chroot** build has completed, go to the instructions in this document, "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Set 8: Build ASSIST After a Full Release 3.0 Build in a Cross Environment on a non-3B2 Computer

Step 1: Follow instructions 1 through 8 in Section 7 of the *Source Code Provision Release Notes.*

Step 2: Change directory to **$ROOT** by executing the following command.

cd $ROOT

Step 3: Load the 9-track magnetic tape on which the ASSIST source code was delivered onto drive 0 of the host machine.

Step 4: Install the ASSIST source by executing

cpio −icdumvB < /dev/rmt/0m

Step 5: Change to the directory containing the ASSIST source code by executing

cd usr/src

Step 6: Build the ASSIST command by executing

./:mkcmd assist

Set 9: Build ASSIST Only in the Standard Root Directory on a 3B2 Computer Running Release 3.0

Step 1: Log in as **root**

Step 2: Install the binary C Programming Language Utilities Issue 4, and the Advanced Programming Utilities Issue 1, using the **sysadm installpkg** utility.

Step 3: Change directory to the **root** (/) directory by executing

cd /

Step 4: Check to see if **libcurses.a** exists by executing

ls −l usr/lib/libcurses.a

Step 5: If **libcurses** exists go to Step 6. If not, install the Terminal Information Utilities that came with your UNIX System V Release 3.0 binary system, using the **sysadm installpkg** utility.

Step 6: Place the Source Code Provision tape in the cartridge tape drive.

Step 7: Install **:mkcmd** from the Source Code tape by executing

ctccpio –idumvT /dev/rSA/ctape1 usr/src/:mkcmd

Step 8: Place the ASSIST cartridge tape in the tape drive.

Step 9: To install the ASSIST source code, enter the following command.

ctccpio –idumvT /dev/rSA/ctape1

Step 10: Change to the directory containing the source.

cd usr/src

Step 11: To build ASSIST, execute the following

./:mkcmd assist

Set 10: Build ASSIST Only in chroot on a 3B2 Computer Running Release 3.0

Step 1: Create the directory **/usr/***new_root* where *new_root* is a name you choose. **/usr/***new_root* will be used as the **root** directory for the installation and build of the Source Code Provision.

mkdir /usr/*new_root*

Step 2: If the System Administration Utilities package has been installed on your system, go to Step 3. If it has not, install it now using the **sysadm installpkg** utility.

Step 3: Change directory to the standard **root** directory.

cd /

Step 4: Copy the following directories from the standard **root** to the new **root** directory.

find bin usr/bin usr/lbin dev usr/admin etc \<CR>
usr/options lib usr/lib shlib usr/include \<CR>
−print | cpio −pudm /usr/*new_root*

Step 5: Change directory to the new **root** directory.

cd /usr/*new_root*

Step 6: Change **root** to the new **root**.

chroot . bin/sh

Step 7: Create the following directories under your new **root** directory.

mkdir install tmp usr/tmp

Step 8: Install the C Programming Language Utilities Issue 4 and the Advanced Programming Utilities Issue 1. Use the **sysadm installpkg** utility.

Step 9: Create the directory *cross_root* under the new **root**. */cross_root* will be used as the **root** of your cross environment.

mkdir *cross_root*

Step 10: Place the Source Code Provision cartridge tape in the tape drive.

Step 11: Change directory to the cross-environment directory.

cd */cross_root*

Step 12: Install the **libcurses** library from the Source Code Provision tape by executing:

ctccpio −idumvT /dev/rSA/ctape1 usr/src/:mkcmd \<CR>
usr/src/:mkhead usr/src/:mksyshead usr/src/:mklib \<CR>
usr/src/head/* usr/src/uts/3b2/sys/* usr/src/lib/\<CR>
libcurses/* usr/src/cmd/cpset.c

Step 13: Place the ASSIST cartridge tape in the tape drive.

Step 14: Install the ASSIST source code by executing

ctccpio –idumvT /dev/rSA/ctape1

Step 15: To create and set the necessary environment execute the following:

SRC=/*cross_root***/usr/src**
export SRC
cd /*cross_root*
mkdir lib usr/include usr/include/sys bin usr/bin etc usr/lib
cd $SRC
./:mkhead
./:mksyshead
ROOT= /*cross_root*
export ROOT
PFX=""
INC=/usr/include
INCSYS=/usr/include/sys
AS=as
CC=cc
LD=ld
export PFX INC INCSYS AS CC LD

Step 16: To build the **libcurses** library and ASSIST, execute the following:

cd $SRC
./:mkcmd cpset
cp $ROOT/usr/bin/cpset /usr/bin/cpset
./:mklib libcurses (This build should take 2 to 4 hours.)
./:mkcmd assist (This build should take 1 to 2 hours.)

If you want to incorporate ASSIST into the standard locations after the **chroot** build has completed, go to the instructions in this document "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Set 11: Build ASSIST Only in chroot on a 3B2 Computer Running Release 2.1 or 2.0.4

Step 1: Follow instructions 1 through 12 in Section 6 of the *Source Code Provision Release Notes.*

Step 2: Place the Source Code Provision cartridge tape in the cartridge tape drive.

Step 3: Install the **libcurses** library from the Source Code Provision tape by executing:

ctccpio -idumvT /dev/rSA/ctape1 usr/src/:mkcmd \<CR>
usr/src/:mkhead usr/src/:mksyshead usr/src/:mklib \<CR>
usr/src/head/* usr/src/uts/3b2/sys/* usr/src/lib/\<CR>
libcurses/* usr/src/cmd/cpset.c

Step 4: Place the ASSIST cartridge tape in the tape drive.

Step 5: Install the ASSIST source code by executing

ctccpio –idumvT /dev/rSA/ctape1

Step 6: To create and set the necessary environment execute the following:

SRC=/*cross_root***/usr/src**
export SRC
cd /*cross_root*
mkdir lib usr/include usr/include/sys bin usr/bin etc usr/lib
cd $SRC
./:mkhead
./:mksyshead
ROOT=/*cross_root*
export ROOT
PFX=""
INC=/usr/include
INCSYS=/usr/include/sys

```
AS=as
CC=cc
LD=ld
export PFX INC INCSYS AS CC LD
```

STEP 7: To build the **libcurses** library and ASSIST, execute the following:

```
cd $SRC
./:mkcmd cpset
cp $ROOT/usr/bin/cpset /usr/bin/cpset
./:mklib libcurses      (This build should take 2 to 4 hours.)
./:mkcmd assist         (This build should take 1 to 2 hours.)
```

If you want to incorporate ASSIST into the standard locations after the **chroot** build has completed, go to the instructions in this document "Incorporating ASSIST into Standard Locations on a 3B2 Computer After a **chroot** Install and Build," which follow Set 11.

Incorporating ASSIST into Standard Locations on a 3B2 Computer After a chroot Install and Build

If you have built ASSIST in a location other than the standard root directory, the following instructions tell you how to incorporate ASSIST into the standard root directory.

Step 1: Type **ctrl-d** to leave the **chroot** environment.

Step 2: Copy **assist** and **astgen** to the standard location in the standard root directory by executing

cp /*new_root*/**usr/bin/assist /usr/bin**
cp /*new_root*/**usr/bin/astgen /usr/bin**

Step 3: Change directory to the **chroot** directory that contains the ASSIST library by executing

cd /*new_root*/**usr/lib/assist**

Step 4: Create a new ASSIST library directory by executing

mkdir /usr/lib/assist

Step 5: Copy the ASSIST library into the standard locations by executing

find . –print | cpio –pdumv /usr/lib/assist

Software Tips

Using ASSIST On Terminals with User-Programmable Function Keys

If your users have terminals that have programmable function keys, you may find that the function keys will work with some of your application packages, and then not work with others.

Initializing programmable function keys is complex. Your function keys must be initialized to their default values for ASSIST to use them. These default values are sometimes changed by software application packages that make the function keys work by clearing the function keys' original values and substituting their own values. It was found through discussions with ASSIST users that some people initialize their function keys themselves and do not want ASSIST to clear or change them.

To make the programmable function keys work with ASSIST, the default values for the keys must be re-initialized on the function keys before ASSSIST is run. In general, information on the default values for function keys and how to reset those values is described in the user manual for the terminal.

Depending on the terminal, users may be able to use the **tput** command to re-initialize their function keys as follows:

- If you have System V, Release 3.0, type the command
  tput init
- If you have System V, Release 2.0, type the commands
  tput is1
  tput is2
  tput is3

If these **tput** commands work for your terminals, then, as the system administrator, you have the option of setting ASSIST so that it automatically clears function keys and re-initializes function keys for all users of your system. Some people might dislike this, so it is suggested that you ask everyone involved before you make any changes.

To set **assist** and **astgen** to automatically re-initialize the function keys, you will add the **tput** instruction(s) to the shell scripts **/usr/bin/assist** and **/usr/bin/astgen**. Follow the steps below that are correct for the version of the UNIX System on your machine.

- UNIX System V, Release 3.0
  - Read in the file **/usr/bin/assist** using any UNIX System editor.
  - Insert the following line at the beginning of the file:

    tput init

  - Your file should now read as follows:

    ```
    tput init
    #ident      "@(#)setup:assist.sh    1.5"
    export ASSISTBIN ASSISTLIB
    ASSISTBIN=/usr/lib/assist/bin
    ASSISTLIB=/usr/lib/assist/lib
    exec $ASSISTBIN/msetup $*
    ```

  - Write the file.
  - Read in the file **/usr/bin/astgen** using any UNIX System editor.
  - Insert the following line at the beginning of the file:

    tput init

  - Your file should now read as follows:

    ```
    tput init
    #ident      "@(#)setup:astgen.sh    1.3"
    export ASSISTBIN ASSISTLIB
    ASSISTBIN=/usr/lib/assist/bin
    ASSISTLIB=/usr/lib/assist/lib
    exec $ASSISTBIN/tsetup $*
    ```

  - Write the file.

- UNIX System V, Release 2.0
  - Read in the file **/usr/bin/assist** using any UNIX System editor.
  - Insert the following lines at the beginning of the file:

```
tput is1
tput is2
tput is3
```

  - Your file should now read as follows:

```
tput is1
tput is2
tput is3
#ident      "@(#)setup:assist.sh    1.5"
export ASSISTBIN ASSISTLIB
ASSISTBIN=/usr/lib/assist/bin
ASSISTLIB=/usr/lib/assist/lib
exec $ASSISTBIN/msetup $*
```

  - Write the file.
  - Read in the file **/usr/bin/astgen** using any UNIX System editor.
  - Insert the following lines at the beginning of the file:

```
tput is1
tput is2
tput is3
```

  - Your file should now read as follows:

```
tput is1
tput is2
tput is3
#ident      "@(#)setup:astgen.sh   1.5"
export ASSISTBIN ASSISTLIB
ASSISTBIN=/usr/lib/assist/bin
ASSISTLIB=/usr/lib/assist/lib
exec $ASSISTBIN/tsetup $*
```

  - Write the file

Enabling the Graphical Character Set

Horizontal and vertical lines in **assist** and **astgen** are drawn as solid lines if the terminal has a graphical character set and as dashed lines otherwise. If your users report horizontal lines composed of "q"s on the ASSIST menus or forms, then the user has told ASSIST during the terminal checking procedures that the terminal has a working graphical character set, but the terminal either does not have a graphical character set or the graphical character set is not enabled.

Depending on the terminal, your users may have to enable the graphical character set each time they turn on their terminals. The user manual for the terminal should explain how to do this. Also, users can try the **tput** sequences described in the section on programmable function keys.

Remind your users that they can use the command **assist -s** (choice "a") to re-evaluate their terminals' graphical character set.

Fixing Return Codes from vi

Previously, the return code conventions for **vi** were that zero (0) was returned if **vi** executed normally and one (1) was returned if **vi** did not execute normally. ASSIST's interaction with **vi** is based on these criteria. However, the return code conventions for **vi** have recently been changed. **vi** now counts the number of user errors made during a **vi** session and returns that number. If you have made exactly one user error, **vi** returns a one (1) and you exit ASSIST prematurely. You will need to make a change to the ASSIST source code in order to properly interpret the new return code conventions.

In the routine "layout()" in file "control.c", approximately at line 660, replace the 6-line fragment:

```
do {
  if (system(s)==256) {
  fprintf(stderr,"Editor in
  exit(9);
  }
}  while (toolong(tf_n) == 0);
```

by the 3-line fragment:

```
do {
  system(s);
  }  while (toolong(tf_n) == 0);
```

Release Format

The ASSIST Software is available for the AT&T 3B2 Computer. Release 1.0 of the ASSIST Software is available in source.

Source Release Files - Delivery and Administration

All of the ASSIST Release 1.0 source code files are specified below and will reside under the /usr/src/cmd/assist directory. These source code files provide the ability to generate all of the executable files.

ASSIST Source Code Files

/usr/src/cmd/assist/assist.mk
/usr/src/cmd/assist/src/assist.sh

/usr/src/cmd/assist/src/forms/command.c
/usr/src/cmd/assist/src/forms/expand.c
/usr/src/cmd/assist/src/forms/fill_out.c
/usr/src/cmd/assist/src/forms/forms.mk
/usr/src/cmd/assist/src/forms/mecho.c
/usr/src/cmd/assist/src/forms/menu.c
/usr/src/cmd/assist/src/forms/mmuse.h
/usr/src/cmd/assist/src/forms/mmusedefs.h
/usr/src/cmd/assist/src/forms/msupport.c
/usr/src/cmd/assist/src/forms/muse.c
/usr/src/cmd/assist/src/forms/muse.h
/usr/src/cmd/assist/src/forms/musedefs.h
/usr/src/cmd/assist/src/forms/popup.c
/usr/src/cmd/assist/src/forms/read_in.c
/usr/src/cmd/assist/src/forms/support.c
/usr/src/cmd/assist/src/forms/update.c
/usr/src/cmd/assist/src/forms/valid.c

/usr/src/cmd/assist/src/script/callout.c
/usr/src/cmd/assist/src/script/do_line.c
/usr/src/cmd/assist/src/script/mkscript.c
/usr/src/cmd/assist/src/script/mscript.c
/usr/src/cmd/assist/src/script/mscript.mk
/usr/src/cmd/assist/src/script/script.h

/usr/src/cmd/assist/src/script/utilities.c

/usr/src/cmd/assist/src/search/format.c
/usr/src/cmd/assist/src/search/msearch.c
/usr/src/cmd/assist/src/search/msearch.h
/usr/src/cmd/assist/src/search/msearch.mk
/usr/src/cmd/assist/src/search/syn.c

/usr/src/cmd/assist/src/setup/intro.c
/usr/src/cmd/assist/src/setup/msetup.c
/usr/src/cmd/assist/src/setup/msetup.h
/usr/src/cmd/assist/src/setup/setup.mk
/usr/src/cmd/assist/src/setup/term.c
/usr/src/cmd/assist/src/setup/update.c

/usr/src/cmd/assist/lib/forms/Batch.fs
/usr/src/cmd/assist/lib/forms/Cflist.fs
/usr/src/cmd/assist/lib/forms/Compare.fs
/usr/src/cmd/assist/lib/forms/Compile.fs
/usr/src/cmd/assist/lib/forms/Create.fs
/usr/src/cmd/assist/lib/forms/Debug.fs
/usr/src/cmd/assist/lib/forms/Directory.fs
/usr/src/cmd/assist/lib/forms/Edit.fs
/usr/src/cmd/assist/lib/forms/Electr.fs
/usr/src/cmd/assist/lib/forms/Find.fs
/usr/src/cmd/assist/lib/forms/Format.fs
/usr/src/cmd/assist/lib/forms/Look.fs
/usr/src/cmd/assist/lib/forms/Maint.fs
/usr/src/cmd/assist/lib/forms/Perm.fs
/usr/src/cmd/assist/lib/forms/Preproc.fs
/usr/src/cmd/assist/lib/forms/Prog.fs
/usr/src/cmd/assist/lib/forms/Setting.fs
/usr/src/cmd/assist/lib/forms/Sorting.fs
/usr/src/cmd/assist/lib/forms/Stats.fs
/usr/src/cmd/assist/lib/forms/System.fs
/usr/src/cmd/assist/lib/forms/TOP.fs
/usr/src/cmd/assist/lib/forms/ar.fs
/usr/src/cmd/assist/lib/forms/ar.val
/usr/src/cmd/assist/lib/forms/ar_d.fs
/usr/src/cmd/assist/lib/forms/ar_m.fs
/usr/src/cmd/assist/lib/forms/ar_p.fs

/usr/src/cmd/assist/lib/forms/ar_q.fs
/usr/src/cmd/assist/lib/forms/ar_r.fs
/usr/src/cmd/assist/lib/forms/ar_t.fs
/usr/src/cmd/assist/lib/forms/ar_x.fs
/usr/src/cmd/assist/lib/forms/assistwalk.fs
/usr/src/cmd/assist/lib/forms/awk.fs
/usr/src/cmd/assist/lib/forms/bfs.fs
/usr/src/cmd/assist/lib/forms/cal.fs
/usr/src/cmd/assist/lib/forms/cat.fs
/usr/src/cmd/assist/lib/forms/cc.fs
/usr/src/cmd/assist/lib/forms/cccomp.fs
/usr/src/cmd/assist/lib/forms/cctyp.fs
/usr/src/cmd/assist/lib/forms/cd.fs
/usr/src/cmd/assist/lib/forms/chmod.fs
/usr/src/cmd/assist/lib/forms/chmoda.fs
/usr/src/cmd/assist/lib/forms/chmods.fs
/usr/src/cmd/assist/lib/forms/chown.fs
/usr/src/cmd/assist/lib/forms/cmp.fs
/usr/src/cmd/assist/lib/forms/comm.fs
/usr/src/cmd/assist/lib/forms/cp.fs
/usr/src/cmd/assist/lib/forms/cpio.fs
/usr/src/cmd/assist/lib/forms/cpioi.fs
/usr/src/cmd/assist/lib/forms/cpioo.fs
/usr/src/cmd/assist/lib/forms/cpiop.fs
/usr/src/cmd/assist/lib/forms/crypt.fs
/usr/src/cmd/assist/lib/forms/csplit.fs
/usr/src/cmd/assist/lib/forms/cu.fs
/usr/src/cmd/assist/lib/forms/cut.fs
/usr/src/cmd/assist/lib/forms/d_help_c
/usr/src/cmd/assist/lib/forms/d_help_m
/usr/src/cmd/assist/lib/forms/d_help_p
/usr/src/cmd/assist/lib/forms/d_help_t
/usr/src/cmd/assist/lib/forms/date.fs
/usr/src/cmd/assist/lib/forms/dc.fs
/usr/src/cmd/assist/lib/forms/df.fs
/usr/src/cmd/assist/lib/forms/diff.fs
/usr/src/cmd/assist/lib/forms/dircmp.fs
/usr/src/cmd/assist/lib/forms/du.fs
/usr/src/cmd/assist/lib/forms/echo.fs
/usr/src/cmd/assist/lib/forms/ed.fs
/usr/src/cmd/assist/lib/forms/egrep.fs

/usr/src/cmd/assist/lib/forms/env.fs
/usr/src/cmd/assist/lib/forms/fgrep.fs
/usr/src/cmd/assist/lib/forms/file.fs
/usr/src/cmd/assist/lib/forms/find.fs
/usr/src/cmd/assist/lib/forms/forms.mk
/usr/src/cmd/assist/lib/forms/g_help_c
/usr/src/cmd/assist/lib/forms/g_help_m
/usr/src/cmd/assist/lib/forms/g_help_p
/usr/src/cmd/assist/lib/forms/g_help_t
/usr/src/cmd/assist/lib/forms/grep.fs
/usr/src/cmd/assist/lib/forms/id.fs
/usr/src/cmd/assist/lib/forms/kill.fs
/usr/src/cmd/assist/lib/forms/ld.fs
/usr/src/cmd/assist/lib/forms/lex.fs
/usr/src/cmd/assist/lib/forms/ln.fs
/usr/src/cmd/assist/lib/forms/lp.fs
/usr/src/cmd/assist/lib/forms/lpstat.fs
/usr/src/cmd/assist/lib/forms/ls.fs
/usr/src/cmd/assist/lib/forms/lscomp.fs
/usr/src/cmd/assist/lib/forms/lstyp.fs
/usr/src/cmd/assist/lib/forms/mail.fs
/usr/src/cmd/assist/lib/forms/make.fs
/usr/src/cmd/assist/lib/forms/mesg.fs
/usr/src/cmd/assist/lib/forms/mkdir.fs
/usr/src/cmd/assist/lib/forms/mm.fs
/usr/src/cmd/assist/lib/forms/mmcomp.fs
/usr/src/cmd/assist/lib/forms/mmtyp.fs
/usr/src/cmd/assist/lib/forms/mv.fs
/usr/src/cmd/assist/lib/forms/nohup.fs
/usr/src/cmd/assist/lib/forms/nroff.fs
/usr/src/cmd/assist/lib/forms/od.fs
/usr/src/cmd/assist/lib/forms/passwd.fs
/usr/src/cmd/assist/lib/forms/paste.fs
/usr/src/cmd/assist/lib/forms/pg.fs
/usr/src/cmd/assist/lib/forms/pr.fs
/usr/src/cmd/assist/lib/forms/prcomp.fs
/usr/src/cmd/assist/lib/forms/prtyp.fs
/usr/src/cmd/assist/lib/forms/ps.fs
/usr/src/cmd/assist/lib/forms/pu_menu.fs
/usr/src/cmd/assist/lib/forms/pwd.fs
/usr/src/cmd/assist/lib/forms/rm.fs

/usr/src/cmd/assist/lib/forms/rmdir.fs
/usr/src/cmd/assist/lib/forms/sdb.fs
/usr/src/cmd/assist/lib/forms/sdbwalk.fs
/usr/src/cmd/assist/lib/forms/sed.fs
/usr/src/cmd/assist/lib/forms/sh.fs
/usr/src/cmd/assist/lib/forms/shell.lpstat
/usr/src/cmd/assist/lib/forms/size.fs
/usr/src/cmd/assist/lib/forms/sort.fs
/usr/src/cmd/assist/lib/forms/sortc.fs
/usr/src/cmd/assist/lib/forms/sorttyp.fs
/usr/src/cmd/assist/lib/forms/spell.fs
/usr/src/cmd/assist/lib/forms/split.fs
/usr/src/cmd/assist/lib/forms/strip.fs
/usr/src/cmd/assist/lib/forms/stty.fs
/usr/src/cmd/assist/lib/forms/sttyrep.fs
/usr/src/cmd/assist/lib/forms/sttyset.fs
/usr/src/cmd/assist/lib/forms/su.fs
/usr/src/cmd/assist/lib/forms/tail.fs
/usr/src/cmd/assist/lib/forms/taili.fs
/usr/src/cmd/assist/lib/forms/tailn.fs
/usr/src/cmd/assist/lib/forms/umask.fs
/usr/src/cmd/assist/lib/forms/uname.fs
/usr/src/cmd/assist/lib/forms/unixwalk.fs
/usr/src/cmd/assist/lib/forms/uucp.fs
/usr/src/cmd/assist/lib/forms/uucpl.fs
/usr/src/cmd/assist/lib/forms/uucpr.fs
/usr/src/cmd/assist/lib/forms/vi.fs
/usr/src/cmd/assist/lib/forms/viwalk.fs
/usr/src/cmd/assist/lib/forms/wc.fs
/usr/src/cmd/assist/lib/forms/who.fs
/usr/src/cmd/assist/lib/forms/yacc.fs

/usr/src/cmd/assist/lib/scripts/as.build
/usr/src/cmd/assist/lib/scripts/scripts.mk
/usr/src/cmd/assist/lib/scripts/sdb.build
/usr/src/cmd/assist/lib/scripts/t.ascf
/usr/src/cmd/assist/lib/scripts/t.ascontinue
/usr/src/cmd/assist/lib/scripts/t.asending
/usr/src/cmd/assist/lib/scripts/t.ashelp
/usr/src/cmd/assist/lib/scripts/t.asintro
/usr/src/cmd/assist/lib/scripts/t.asmenu

/usr/src/cmd/assist/lib/scripts/t.aspopmenu
/usr/src/cmd/assist/lib/scripts/t.assubs
/usr/src/cmd/assist/lib/scripts/t.sdbbreak
/usr/src/cmd/assist/lib/scripts/t.sdbcontinue
/usr/src/cmd/assist/lib/scripts/t.sdbcore
/usr/src/cmd/assist/lib/scripts/t.sdbending
/usr/src/cmd/assist/lib/scripts/t.sdbexec
/usr/src/cmd/assist/lib/scripts/t.sdbintro
/usr/src/cmd/assist/lib/scripts/t.sdbmulti
/usr/src/cmd/assist/lib/scripts/t.sdbsdbend
/usr/src/cmd/assist/lib/scripts/t.sdbsearch
/usr/src/cmd/assist/lib/scripts/t.sdbsubs
/usr/src/cmd/assist/lib/scripts/t.uncontinue
/usr/src/cmd/assist/lib/scripts/t.unending
/usr/src/cmd/assist/lib/scripts/t.unfiles
/usr/src/cmd/assist/lib/scripts/t.unintro
/usr/src/cmd/assist/lib/scripts/t.unintroend
/usr/src/cmd/assist/lib/scripts/t.unpipes
/usr/src/cmd/assist/lib/scripts/t.unrediri
/usr/src/cmd/assist/lib/scripts/t.unrediro
/usr/src/cmd/assist/lib/scripts/t.unstdio
/usr/src/cmd/assist/lib/scripts/t.unsubs
/usr/src/cmd/assist/lib/scripts/t.viadd
/usr/src/cmd/assist/lib/scripts/t.vicontinue
/usr/src/cmd/assist/lib/scripts/t.vicursor
/usr/src/cmd/assist/lib/scripts/t.videlete
/usr/src/cmd/assist/lib/scripts/t.videmo
/usr/src/cmd/assist/lib/scripts/t.viending
/usr/src/cmd/assist/lib/scripts/t.viintro
/usr/src/cmd/assist/lib/scripts/t.viintroend
/usr/src/cmd/assist/lib/scripts/t.vimanip
/usr/src/cmd/assist/lib/scripts/t.viopts
/usr/src/cmd/assist/lib/scripts/t.visubs
/usr/src/cmd/assist/lib/scripts/t.viundo
/usr/src/cmd/assist/lib/scripts/un.build
/usr/src/cmd/assist/lib/scripts/vi.build
/usr/src/cmd/assist/lib/scripts/vicommands

/usr/src/cmd/assist/lib/search/search.mk
/usr/src/cmd/assist/lib/search/searchlist
/usr/src/cmd/assist/lib/search/unix/matchpairs

/usr/src/cmd/assist/lib/search/unix/search-file

/usr/src/cmd/assist/lib/setup/assistrc
/usr/src/cmd/assist/lib/setup/messages
/usr/src/cmd/assist/lib/setup/setup.mk
/usr/src/cmd/assist/lib/setup/t.altchar
/usr/src/cmd/assist/lib/setup/t.basic
/usr/src/cmd/assist/lib/setup/t.fkeys
/usr/src/cmd/assist/lib/setup/t.standout

astgen Source Code Files

/usr/src/cmd/assist/src/astgen.sh
/usr/src/cmd/assist/src/astgen/bottom.c
/usr/src/cmd/assist/src/astgen/cf_field.c
/usr/src/cmd/assist/src/astgen/cf_global.c
/usr/src/cmd/assist/src/astgen/control.c
/usr/src/cmd/assist/src/astgen/dhelp.c
/usr/src/cmd/assist/src/astgen/editval.c
/usr/src/cmd/assist/src/astgen/err_mess.c
/usr/src/cmd/assist/src/astgen/field_type.c
/usr/src/cmd/assist/src/astgen/get_path.c
/usr/src/cmd/assist/src/astgen/m_field.c
/usr/src/cmd/assist/src/astgen/m_global.c
/usr/src/cmd/assist/src/astgen/move.c
/usr/src/cmd/assist/src/astgen/perm_chk.c
/usr/src/cmd/assist/src/astgen/pick_val.c
/usr/src/cmd/assist/src/astgen/select.c
/usr/src/cmd/assist/src/astgen/show_cap.c
/usr/src/cmd/assist/src/astgen/tools.h
/usr/src/cmd/assist/src/astgen/tools.mk
/usr/src/cmd/assist/src/astgen/util.c
/usr/src/cmd/assist/src/astgen/vdefs.h
/usr/src/cmd/assist/src/astgen/vsupport.c
/usr/src/cmd/assist/src/astgen/writefile.c

/usr/src/cmd/assist/lib/astgen/cf_f.help
/usr/src/cmd/assist/lib/astgen/cf_g.help
/usr/src/cmd/assist/lib/astgen/cfinfo
/usr/src/cmd/assist/lib/astgen/ftype.help
/usr/src/cmd/assist/lib/astgen/m_f.help

/usr/src/cmd/assist/lib/astgen/m_g.help
/usr/src/cmd/assist/lib/astgen/menuinfo
/usr/src/cmd/assist/lib/astgen/perm.help
/usr/src/cmd/assist/lib/astgen/tools.mk
/usr/src/cmd/assist/lib/astgen/top.help
/usr/src/cmd/assist/lib/astgen/vals.help

Storage requirements for the major source directories are summarized in the following tables:

Menu Interface Storage Requirements	
Directory	**Size(512-byte blocks)**
/usr/src/cmd/assist/src/forms	437
/usr/src/cmd/assist/src/script	145
/usr/src/cmd/assist/src/search	43
/usr/src/cmd/assist/src/setup	80
/usr/src/cmd/assist/lib/forms	769
/usr/src/cmd/assist/lib/scripts	298
/usr/src/cmd/assist/lib/search	102
/usr/src/cmd/assist/lib/setup	25
Total	1899

Menu/Form Generator Tool Storage Requirements	
Directory	**Size(512-byte blocks)**
/usr/src/cmd/assist/src/astgen	443
/usr/src/cmd/assist/lib/astgen	70
Total	513

In total, the ASSIST source package will use no more than 2500 blocks of space. Once the system has been installed, the storage requirements can be reduced by approximately 1000 blocks. This reduction results from two things: some of the data files are packed, and you can remove the source files from the source tree. To do this, type the following:

```
cd /usr/src/cmd
rm -fr assist
```

Documentation Available

The list below includes the ASSIST documents available and their Select Code number. Refer to these numbers when you order the documents.

- *ASSIST Software Release 1.0 Development Tools Guide* (307-235)
- *ASSIST Software Release 1.0 Source Code Provision Release Notes* (307-236)
- *ASSIST Software Release 1.0 User's Guide* (307-234)

How to Order Documents

Additional copies of these documents or other UNIX System documents can be ordered by calling AT&T Customer Information Center (CIC):

1-800-432-6600 (toll free within the continental United States)

1-317-352-8556 (outside the continental United States)

or by writing to:

AT&T Customer Information Center
Customer Service Representative
P. O. Box 19901
Indianapolis, Indiana 46219