AIX 6 Basics

(Course Code AU13)

Instructor Exercises Guide with Hints

ERC 10.0

IBM Certified Course Material

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX®	AIX 5L™	Common User Access®
MVS™	OS/2®	pSeries®
System p™	System p5™	400®

PS/2® is a trademark or registered trademark of Lenovo in the United States, other countries, or both.

Adobe is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX® is a registered trademark of The Open Group in the United States and other countries.

Linux® is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

February 2008 Edition

Contents

Trademarks

The reader should recognize that the following terms, which appear in the content of this training document, are official trademarks of IBM or other companies:

IBM® is a registered trademark of International Business Machines Corporation.

The following are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX®	AIX 5L™	Common User Access®
MVS™	OS/2®	pSeries®
System p™	System p5™	400®

PS/2® is a trademark or registered trademark of Lenovo in the United States, other countries, or both.

Adobe is either a registered trademark or a trademark of Adobe Systems Incorporated in the United States, and/or other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX® is a registered trademark of The Open Group in the United States and other countries.

Linux® is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Instructor Exercises Overview

The objective of the AIX Basics exercises is to have students understand and successfully perform basic user functions in AIX. There are no system administration activities performed or discussed in the exercises. **Known Hardware/Software Problems** sections are provided with each exercise. These sections alert you to known problems or common mistakes that are inherent to the exercise.

Be sure you understand the equipment configuration for your teach site before introducing the exercises. For the purpose of these exercises, it will be assumed that the login user names are **team01** through **team05**. The passwords should be the same as the user names. Check the configured systems before the start of class for the login user names and passwords. Should this not be the case, contact the classroom administrator to have these user names and passwords added or find out what login names and passwords are available. This information will be introduced in the first exercise. The students will not be introduced to nor use the **root** login, but you may have to log in as root to configure or fix a system problem. It is assumed that **root**'s password is **`ibmaix`**. Check for root's password before the start of class. If this is not **root**'s password, contact the classroom administrator to either obtain root's password or ensure that hardware/software support is available to you during the class should any system administrative type functions be required.

Due to the variety of classrooms where this class may be offered, the equipment available in any one classroom may be significantly different from another. Some classrooms will have a complete computer system for each student. The World Wide standard is to have one graphics terminal (LFT) and one network-connected PC running an AIX emulation program per IBM System p server. The recommendation is no more than two students per graphics terminal. However, some classrooms may have as many as 10-15 network-connected PCs with terminal emulation software connecting to a single IBM System p server or LPAR.

Because of these differences, you will need to exercise judgement when students do the exercises. There may be times when students happen to log in with the same user name as someone else on the same system. If this happens, results they see during the exercises may be confusing.

Exercise 2 requires that the student machines have access to the Internet-based AIX Information Center, or an AIX system has the Information Center installed and remotely accessible. The **`man`** pages

should already be installed. Check to ensure this has been done. If not, contact the classroom administrator to see if it can be installed. If it is not, the exercise cannot be performed. Also ensure that the file **/usr/share/man/whatis** is on each system. If it does not exist, log in as root and run the command **`catman -w`**. This will create the **`whatis`** database.

There are appendices units and exercises that provide exercises for the AIXwindows environment. They are options and the students can choose to do them if time and the classroom environment permits. They require a LFT connected to the IBM System p machine, with keyboard and mouse, and the AIXwindows and CDE environment installed.

The **Exercise Review/Wrap-up** section at the end of each exercise in this guide is meant to be conducted by you as an exercise review at the end of each exercise session and before the beginning of the next unit. The students do NOT have this section in their exercise notebook. These are “suggested” review questions. To add to the exercise review, keep track of questions and common problems students had during the exercise. Use the board to cover these common problems. Simply state that these are a few of the common problems seen in all classes. Do not single out any one team or student. Keep it a group discussion.

Exercises Configuration

See the Lab Setup Guide for complete details on hardware and software configuration.

Exercises Description

None of the exercises, EXCEPT Exercises 4 and 5 are dependent on the preceding exercise being successfully completed. It is assumed, however, that you understand the commands and concepts from each exercise as these commands and concepts are carried over to the follow-on exercises.

Each exercise in this course is divided into sections as described below. Select the section that best fits your method of performing exercises. You may select to use a combination of these sections as appropriate.

Exercise Instructions - This section contains what it is you are to accomplish. There are no definitive details on how to perform the tasks. You are given the opportunity to work through the exercise given what you learned in the unit presentation, utilizing the unit Student Notebook, your past experience, and maybe a little intuition.

Exercise Instructions with Hints - This section is an exact duplicate of the **Exercise Instructions** section except that in addition, specific details and hints are provided to help step you through the exercise. Using the **Exercise Instructions** section along with the **Exercise Instructions with Hints** section can make for a rewarding combination providing you with no hints when you do not want them and hints when you need them. When there is more than one way to do a command, we show you both ways with an **-OR-** between possible solutions.

Optional Exercises - This section provides additional practice on a particular topic. Specific details and hints are provided to help step you through the **Optional Exercises**, if needed. Not all exercises include **Optional Exercises**.

Solutions - This section provides at least one solution to questions strategically placed in some exercises. Where applicable the solutions have been provided at the end of the **Exercise Instructions with Hints** section. Note: These are NOT the solutions to the exercises as those are provided in the **Exercise Instructions with Hints**.

Text highlighting

The following text highlighting conventions are used throughout this book:

Bold	Identifies file names, file paths, directories, user names, and principals.
Italics	Identifies links to Web sites, publication titles, and is used where the word or phrase is meant to stand out from the surrounding text.
`Monospace`	Identifies attributes, variables, file listings, SMIT menus, code examples of text similar to what you might see displayed, examples of portions of program code similar to what you might write as a programmer, and messages from the system.
`Monospace bold`	Identifies commands, daemons, menu paths, and what the user would enter in examples of commands and SMIT menus.
`<text>`	The text between the < and > symbols identifies information the user must supply. The text may be normal highlighting, **bold** or `monospace`, or **`monospace bold`** depending on the context.

Exercise 1. Using the System

(with Hints)

Estimated Time

00:30

What This Exercise Is About

The purpose of this exercise is to become familiar with AIX command syntax and basic commands.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Log in to an AIX system and change passwords
- Execute basic commands
- Use the **wall** and **write** commands to communicate with other users
- Use keyboard control keys to control command line output

Introduction

When executing commands on the command line, use the **Enter** key on the graphics keyboard not the **Ctrl/Act** key. If using an ASCII keyboard use the **Return** key not the **Send** key. Use of the **Ctrl/Act** or **Send** keys can cause unpredictable results. When correcting a typographical error on the command line, use the **Backspace** key not the **arrow** keys.

Common Student Problems

Instruction 7 on page 5 - If a student uses the **finger** command and sees **???** in the Name field, explain that this means that *OPTIONAL* user information was not added to the user profile when created (that is, the /**etc**/**passwd** file). The command executed correctly.

Instruction 11 on page 5 - If the **clear** command does not work on the ASCII terminals, check to see that the TERM variable is correctly set.

Known Hardware/Software Problems

Due to the variety of classrooms where this class may be offered, the equipment available in any one classroom may be significantly different from another. Some classrooms will have a complete computer system for each student and possibly a PC running AIX in emulation mode attached to each IBM System p system. Other classrooms may have as many as 10-15 Xstations and 3151 terminals attached to a single, larger System p machine. If this is the case be aware that if you log in using the same userid as someone else, results you see during the exercise may be confusing.

Even more common these day is the use of remote lab equipment with the students using telnet protocols to connect to the remote lab machines. Once again, it is common to have several students on the same machine and they need to be careful to use unique userids.

If the tool being used for ascii terminal emulation is **PuTTY**, then the student will need to be taught some PuTTY basics. The most common error is not changing the default protocol from ssh to telnet. The second most common change made is the definition of the backspace key. In the category tree on the left side of the PuTTY Configuration panel, they need to select the Keyboard item under Terminal. On the resulting panel they need to change the backspace definition to “Control-H” and then apply the change. If they do not do this before having PuTTY launch a telnet session, they will have to modify the existing session. To do this, they would right-click on the title bar of the telnet session; and then, select “Change Settings” in the resulting menu. This will give them the PuTTY Configuration panel, in which they can make the necessary change. **If they do not redefine the backspace key, the lab will not work as designed!**

The **`mail`** command is designed to work in a DNS environment. Since our machine environment for class uses the **/etc/hosts** file rather than DNS, it may take around 5 minutes to receive mail even when it was sent from your own system. In our classroom environments, this can be a problem. The **/etc/hosts** file will need to show entries such as: `9.19.98.1 sys1 sys1` in order to work around this situation. The course *Lab Setup Guide* indicates this as a necessary step in setting up the exercises for this course.

For step 16, if the student does not have a partner on the same machine to communicate with, see if the student can play both ends. This assumes that there is an extra terminal or at least a PC with a telnet connection, for the student to log in as a different user. If there is no way to have two sessions simultaneously, the student could “write”

to themself, but this is not very clear (it looks like they are just echoing what they write).

For step 17, if the student is not on a local terminal, the **<Ctrl-s>** may not be accepted. Even if it is accepted the output may already be queued for display before the control sequence is accepted.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Logging In / Changing Passwords

__ 1. Log in to the system with the user name and password provided by your instructor. It should be a user name such as **teamxx** where **xx** is a double digit number like **01, 02** and so forth.

The very first time you log in with your new user name, you will be prompted to change your password. Keep the password the same as your login name. The passwords you supply on the command line will not be displayed.

```
» login: teamxx (at the login prompt)
Password: teamxx (default password same as user name)
You are required to change your password. Please choose
a new one.
teamxx's New password: teamxx (keep it the same for now)
Enter new password again: teamxx
```

__ 2. Verify that the password has been set by logging out and back in.

```
» $ exit
login: teamxx
Password: (key in your new password)
```

Basic Commands

__ 3. Display the system's date.

```
» $ date
```

__ 4. Display the whole calendar for the year 2007.

```
» $ cal 2007
```

__ 5. Display the month of September for the year 1752. Notice anything peculiar about September? _____

```
» $ cal 9 1752
```

__ 6. Display the month of January for the years 1999 and 99. Are 1999 and 99 the same? _____

» $ **cal 1 1999**

» $ **cal 1 99**

__ 7. There are two commands that will display information about all users currently on the local system. Display who is currently logged in on your system. Check to see when they logged in.

» $ **who**
-OR

» $ **finger**

__ 8. Display just your login name.

» $ **who am i**

__ 9. Use **banner** to display `Out to Lunch`.

» $ **banner Out to Lunch**

__ 10. Use the **echo** command to write the character string `Out to Lunch` to your display.

» $ **echo Out to Lunch**

__ 11. Use the **clear** command to clear your screen.

» $ **clear**

Send and Receive Mail

__ 12. Send a note to yourself using the **mail** command. Provide a subject but ignore the carbon copy prompt.

» $ **mail teamxx** (where **teamxx** is your login name)
Subject: **A reminder to myself**
The meeting starts at 10:00.
<Ctrl-d> (**<Ctrl-d>** must start on a new line)
Cc: (enter to bypass this option)

__ 13. Start the **mail** process and list the message in your mailbox. Read your message, save it, and quit the **mail** program. To list a brief summary of **mail** subcommands, type **?** at the **mail** prompt.

» $ **mail**
? **t** (you can also use 1 if preferred)
? **s**
"/home/teamxx/mbox" [New file] (You will see this message)
? **q**

__ 14. Access your mail and delete the message you saved in your personal mailbox. Exit the **mail** program. If there is more than one person logged in on your system, practice sending mail to each other.

» $ **mail -f**
? **d**
? **q**

Communicating with Other Users

__ 15. Send a note to all users on the system indicating that you have almost completed this exercise.

» $ **wall I have almost completed this exercise**

__ 16. Pair up with someone on your system to coordinate this exercise. Open a line of communication to send a message to your partner, **teamyy**. Let teamyy know that you are waiting for a response. **teamyy** should then reply and let you know that they have nothing else to say. End of conversation.

» $ **write teamyy**
I need to see you
o

» $ **write teamxx**
I am too busy at the moment
oo
<Ctrl-d>

» Enter a **<Ctrl-d>** to end your conversation after seeing the **oo** from **teamyy**.

Keyboard Tips

To get some practice temporarily stopping, starting, and terminating the scrolling of command output, use the **banner** command to banner the letters of the alphabet in order to generate multiple lines of output.

__ 17. Using **banner**, display the alphabet separating each character with a space. As output is scrolling to your display, temporarily stop the output. Resume the scrolling.

» $ **banner a b c d e f g h i j k l m n o p q r s t u v w x y z**
<Ctrl-s> (temporarily stops scrolling)
<Ctrl-q> (resumes scrolling)
<Ctrl-c> (terminates the current command)

__ 18. Repeat the **banner** command used in the previous step, typing only the first five letters of the alphabet, but DO NOT press **Enter**. Erase your input using **<Ctrl-u>**. Now have the **banner** command display the phrase `End of Exercise`. This time if you make a typing mistake while keying this command, use the **Backspace** key to correct the command line.

```
» $ banner a b c d e
  $ <Ctrl-u>
  $ banner End of Exercise
```

__ 19. Log off the system.

```
» $ <Ctrl-d>
```

END OF EXERCISE

Exercise Solutions

__ 5. Display the month of September for the year 1752. Notice anything peculiar about September? _____

» **Answer**: This was an adjustment made by Pope Gregory to bring the calendar back in sync with the Earth's rotation, causing much upheaval among the population which felt that he had taken away eleven days of their lives!

__ 6. Display the month of January for the years 1999 and 99. Are 1999 and 99 the same? _____

» **Answer**: No, they are not the same. The year is taken literally. You must be specific as to the century as well.

Exercise Review/Wrap-up

1. In instruction 9 on page 5 and instruction 10 on page 5 you used the **`banner`** and **`echo`** command to display the message that you are out to lunch. Should this really be done?

   Answer: Probably not. It would be better to actually log off the system for security reasons.

2. What key stroke must you press to start and stop long files or messages from scrolling off the screen before you can read them?

   Answer: **`<Ctrl-s>`** and **`<Ctrl-q>`**. You will learn a better way of doing this in a later unit.

3. What are the three ways to log off the system?

   Answer: **`exit, logout, <Ctrl-d>`**. In later discussions, you will find why **`<Ctrl-d>`** may not always work.

Exercise 2. AIX 6.1 Documentation

(with Hints)

Estimated Time

00:45

What This Exercise Is About

The purpose of this exercise is to give the students the opportunity to explore and experiment with the **man** command and with the AIX 6.1 online documentation.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Execute the **man** command
- Start a Web browser to access the online documentation

Introduction

In this exercise, you will first use the **man** command from the command line. This part of the exercise can be performed in either graphics mode or ASCII mode.

In the second part of the exercise, you will use a Web browser to access AIX 6.1 online documentation.

Common Student Problems

Be sure to have the students read the Introduction to the machine exercise.

In order to run this exercise, students will need a Web browser with access to the Internet, or an AIX machine set up as a documentation server. If necessary, consult the Lab Setup Guide for instructions on how to configure a documentation server.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

man Pages

__ 1. Log in to the system with the user name and password provided by your instructor.

__ 2. Bring up the **man** pages for the **man** command. Read the text that follows to obtain a better understanding of the functionality of the **man** command.

Remember to use the space bar to go forward one screen and the return key to go forward one line. Press the **b** key to go back one screen. When you have read enough, exit **man** using the **q** key or **<Ctrl-c>**.

» $ **man man**

» **<Ctrl-c>** or **q**

__ 3. Using the **man** command, search on the keyword `calendar`. From the list produced, find the command that displays a calendar.

» $ **man -k calendar**

__ 4. Having found the **cal** command from the previous step, use **man** without any options to obtain the correct syntax of the command.

» $ **man cal**

AIX Information Center

__ 5. Start up a Web browser and access the online documentation. Your instructor will tell you whether to use the Internet site or a local AIX system configured as a documentation server.
The URL for the Internet site is:
http://publib.boulder.ibm.com/infocenter/pseries/v6r1/index.jsp

__ 6. Click the **AIX Information** link in the left frame.

» Click **AIX Information**

__ 7. In the right frame of your Web browser, click the link entitled *Operating System and Device Management*.

» Click **Operating System and Device Management**

__ 8. Select one or two of the topics displayed in the right frame.

» On the page, you will see various topics and links to the appropriate HTML pages. Click a few links to view the pages. Use the left and right arrow

buttons located in the upper right corner of the frame to navigate around your pages.

__ 9. Now, suppose you do not know what document to look in for the information you require. Use the search function in the Information Center to find information on the **wc** command.

» In the search box, enter the string **wc command** and press **Enter** or click **Go**. In the right frame, you will see the results of your search. The words you searched on will be highlighted in each document. Select one of the top documents and scroll through it. Use the left arrow in the upper right corner to go back to the search results and select another document.

__ 10. Use the **Search Scope** function to narrow your search. Change the search scope for the previous search to only include the Commands Reference.

» Click **Search Scope:** located next to the Search box.

» A pop-up box will appear allowing you to define a custom search scope. Select the radio button entitled **Search only the following topics** and click the **New** button.

» In the box entitled **List Box**, type an appropriate name for your new scope, such as “man pages.” In the main window, click the “**+**” next to **AIX Information**, and select the box next to **Commands**. A checkmark will appear in the box. Click **Ok** to save your scope.

» You should now have your new search scope highlighted. Click **Ok** to select it.

» You will see that the text next to **Search Scope:** now says **man pages**. Click **Go** next to the search box to rerun your search.

» You will now see your results in the left frame. Select a few of the results and browse the pages.

__ 11. Use any extra time you have to explore other documents available in the AIX Information Center.

__ 12. Exit your Web browser.

» Click **File**

» Click **Exit**.

END OF EXERCISE

Exercise 3. Files and Directories

(with Hints)

Estimated Time

00:30

What This Exercise Is About

This exercise provides the students with the opportunity to begin working with directories and the files they contain.

What You Should Be Able to Do

After completing this exercise, you should be able to:

- Display the name of the current directory
- Change directories
- Use various options of the **`ls`** command to display information about files and directories
- Create and remove directories
- Create zero-length files

Introduction

In this exercise, you will be using AIX commands to work with directories and files.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

__ 1. If you are not already logged in, log in to the system

» Login: **teamxx**

» teamxx's Password: **teamxx**

__ 2. Using the **pwd** command, verify that you are in your home directory, /**home**/**teamxx**, the directory where you are placed when you first log in.

» $ **pwd**

__ 3. Change your current directory to the root directory (/).

» $ **cd /**

__ 4. Verify that you are in the root directory and then execute both a simple and a long listing of the files in that directory.

» $ **pwd**

» $ **ls**

» $ **ls -l**

__ 5. Issue the **ls** command with the **-a** and the **-R** options. What is the effect of each option?_____ (**Note**: The **ls -R** will provide extensive output. Once you have seen enough, enter the key sequence **<Ctrl-c>** to end the command.)

» $ **ls -a**

» $ **ls -R**

__ 6. Return to your home directory (/**home**/**teamxx**) and list its contents including hidden files.

» $ **cd**

» $ **ls -a**

__ 7. Create a directory in your home directory called **mydir**. Then, issue commands to view a long listing of both your /**home**/**teamxx**/**mydir** and /**home**/**teamxx** directories. What are the sizes of each directory?_____

» $ **mkdir mydir**

» $ **ls -ld /home/teamxx/mydir**

» $ **ls -ld /home/teamxx**

__ 8. Change to the /**home**/**teamxx**/**mydir** directory. Use the **touch** command to create two zero-length files called **myfile1** and **myfile2** in your **mydir** directory.

» $ **cd mydir**

» $ **touch myfile1**

» $ **touch myfile2**

__ 9. Issue the command to view a long listing of the contents of your **mydir** directory. What are the sizes of **myfile1** and **myfile2**?_____ View the long listing again, this time also displaying the i-node numbers of the files. What are the i-node numbers for the files? _____

» $ **ls -l**

» $ **ls -li**

__ 10. Change back to your home directory and issue the **ls -R** command to view your directory tree.

» $ **cd**

» $ **ls -R**

__ 11. Use the **istat** command to view i-node information on your **mydir** directory. Why may the “Last Accessed” date be more current than the other two dates?

» $ **istat mydir**

__ 12. Use the **rmdir** command to remove the **mydir** directory. Does it work? _____ You will note that the **rmdir** command cannot remove a non-empty directory. To do that, you will need to issue a command that we will learn in the next unit, **rm -r**.

» $ **rmdir mydir**

» $ **rm -r mydir**

END OF EXERCISE

Exercise Solutions

__ 5. Issue the **ls** command with the **-a** and the **-R** options. What is the effect of each option?_____ (**Note**: The **ls -R** will provide extensive output. Once you have seen enough, enter the key sequence **<Ctrl-c>** to end the command.)

» The **-a** command option displays all hidden files (files that begin with a "**.**"). The **-R** command option displays files recursively in a directory structure.

__ 7. Create a directory in your home directory called **mydir**. Then, issue commands to view a long listing of both your /**home**/**teamxx**/**mydir** and /**home**/**teamxx** directories. What are the sizes of each directory?_____

» The file size is the number just before the last modified date on the report line. It is probably 512.

__ 9. Issue the command to view a long listing of the contents of your **mydir** directory. What are the sizes of **myfile1** and **myfile2**?_____ View the long listing again, this time also displaying the i-node numbers of the files. What are the i-node numbers for the files? _____

» The files sizes should be 0.

» The i-nodes number is the first field on each **ls -li** report line. The numbers will likely vary from system to system.

__ 11. Use the **istat** command to view i-node information on your **mydir** directory. Why may the "Last Accessed" date be more current than the other two dates?

» The "Last Accessed" date will be updated any time the directory is viewed. The other dates are updated when the directory or its i-node structure is changed.

__ 12. Use the **rmdir** command to remove the **mydir** directory. Does it work? _____ You will note that the **rmdir** command cannot remove a non-empty directory. To do that, you will need to issue a command that we will learn in the next unit, **rm -r**.

» The **rmdir** fails with an error message: `Directory mydir is not empty`.

Exercise Review/Wrap-up

1. Review the difference in output from **`ls, ls -a, ls -R`**, and **`ls -i.`**
2. Further explain the last step of the exercise, step 12. Students will learn more about the **`rm -r`** command in the next unit and exercise.

Exercise 4. Using Files

(with Hints)

Estimated Time

00:45

What This Exercise Is About

In this exercise, students use a number of AIX commands to manipulate files.

What You Should Be Able to Do

After completing this exercise, a student should be able to:

- Copy, move, rename, link, and remove files
- Display the contents of a file
- Print a file

Introduction

In this exercise you will be using AIX commands to manipulate ordinary files and directories using the commands discussed in lecture.

Common Student Problems

Make sure students are aware of their current directory while performing the various steps. If students lose track of where they are in the file tree structure, some steps will appear to be broken. Have them use `pwd` frequently to check their current directory.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Checking Your Environment

__ 1. If not already logged in, log in to the system.

» Login: **teamxx**

» teamxx's Password: **teamxx**

__ 2. Using **pwd**, verify that you are in your home directory, /**home**/**teamxx**, the directory where you are placed when you first log in.

» $ **pwd**

__ 3. List the contents of your home directory **(**/**home**/**teamxx**), including hidden files.

» $ **ls -a**

Working with Files

__ 4. Look at the contents of the /**etc**/**hosts** and /**etc**/**profile** files. Use the commands **cat**, **pg**, and **more** to see how each command handles the output.

» $ **cat /etc/hosts**

» $ **cat /etc/profile**

» $ **pg /etc/hosts**

» $ **pg /etc/profile**

» $ **more /etc/hosts**

» $ **more /etc/profile**

__ 5. Copy the file /**usr**/**bin**/**cat** into your current (home) directory.

» $ **cp /usr/bin/cat /home/teamxx**

-OR-

» $ **cp /usr/bin/cat .**

__ 6. Copy the file /**usr**/**bin**/**cal** into your current (home) directory.

» $ **cp /usr/bin/cal /home/teamxX**

-OR-

» $ **cp /usr/bin/cal .**

__ 7. List the files in your current directory. You should see the two you just copied.

```
» $ ls
```

Creating and Manipulating Directories

__ 8. Create a subdirectory in your home directory called **myscripts**.

```
» $ mkdir myscripts
```

__ 9. Move and rename the two files that you just copied to your home directory (**cat** and **cal**) into your new subdirectory. Name the new files **mycat** and **mycal** respectively.

```
» $ mv cat myscripts/mycat
```

```
» $ mv cal myscripts/mycal
```

__ 10. Make the new subdirectory, **myscripts**, your current directory.

```
» $ cd myscripts
```

__ 11. List the contents of the directory to make sure that the files were copied.

```
» $ ls
```

__ 12. Use the **mycat** command in your **myscripts** directory to look at the contents of the **.profile** file in your home directory.

```
» $ mycat ../.profile
```

-OR-

```
» $ mycat /home/teamxx/.profile
```

__ 13. Make your home directory the current directory.

```
» $ cd
```

__ 14. Create another subdirectory in your home directory called **goodstuff**.

```
» $ mkdir goodstuff
```

__ 15. Copy a file called /**etc**/**profile** into the new directory, and name the new file **newprofile**.

```
» $ cp /etc/profile goodstuff/newprofile
```

__ 16. Use the **cat** command to look at the file. Hard to read? Try the **pg** command.

```
» $ cat goodstuff/newprofile
```

```
» $ pg goodstuff/newprofile
```

__ 17. The filename, **newprofile**, is too long to keep typing. Change its name to **np**. List the contents of the **goodstuff** directory to make sure that you have accomplished the task. Use the **cat** command to type out the renamed file.

```
» $ mv goodstuff/newprofile goodstuff/np
```

```
» $ ls goodstuff
```

» $ **cat goodstuff/np**

__ 18. This is a good point to check everything out. Starting from your home directory and working downwards, display a hierarchical tree of your files and subdirectories.

» $ **cd**

» $ **ls -R**

Remove a Directory

__ 19. Ensure you are in your home directory. Remove the **goodstuff** directory. Could you do it? Why or why not?

» $ **pwd**

» $ **rmdir goodstuff**

__ 20. Change to the **goodstuff** directory. Do a listing on the contents of the **goodstuff** directory including any hidden files. Remove the files. Do another listing on the **goodstuff** directory including the hidden files. Notice the . and .. files are still there. The directory is considered “empty” if these are the only two entries left in it. Remove the directory.

» $ **cd goodstuff**

» $ **ls -a**

» $ **rm np**

» $ **ls -a**

» $ **cd ..**

» $ **rmdir goodstuff**

END OF EXERCISE

Optional Exercises

__ 21. Using the **mkdir** command only once, create a directory under the **myscripts** directory named **sports** that has three directories in it named **tennis, basketball,** and **baseball**. Check to be sure the directories were created properly.

```
» $ cd /home/teamxx/myscripts
» $ mkdir -p sports/tennis sports/basketball sports/baseball
» $ ls sports
```

__ 22. Copy the file /**etc**/**motd** into the **tennis** directory and create two files in the **basketball** directory. Leave the **baseball** directory empty. Check to be sure the files were created.

```
» $ cp /etc/motd sports/tennis
» $ touch sports/basketball/myteam
» $ touch sports/basketball/myplayer
» $ ls sports/tennis sports/basketball
```

__ 23. Use the **rm** command to remove the **sports** directory and everything in it.

```
» $ rm -r sports
```

END OF OPTIONAL EXERCISES

Solutions

Following are the solutions for those instructions that included questions:

__ 20. Ensure you are in your home directory. Remove the **goodstuff** directory. Could you do it?

Why not?

» Answer: You should not be able to remove the **goodstuff** directory because it has files in it.

Exercise Review/Wrap-up

1. Review relative and full path name with a copy or move example.
2. Review the difference in output from `ls`, `ls -a`, and `ls -R`.
3. Ask the students what the difference is in output when looking at files with the `cat`, `pg`, and `more` commands.

   Answer: With the `cat` command, if the file is larger than a screen, the file will continue to scroll until it reaches the end. `pg` and `more` format the output to fit the screen and waits for further instructions.
4. Ask students to help you draw on the board a tree structure of how their directory structure looked prior to removing **goodstuff**.

Exercise 5. File Permissions

(with Hints)

Estimated Time

00:45

What This Exercise Is About

The purpose of this exercise is to give the student the opportunity to work with file and directory permissions. A fundamental understanding of basic AIX file ownership and permissions should be a result of performing these exercises.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Manipulate permissions on ordinary files and directories
- Interpret file and directory permission bits
- Display long listing information for files and directories

Introduction

In this exercise, you will be using AIX commands to manipulate AIX file and directory permissions. Understanding the implications of file permissions and ownership and using the commands to change file permissions is necessary to doing additional exercises in this course.

Tips

Make sure you are aware of what directory you are in while performing the various steps. If you lose track of where you are in the exercise, some instructions will appear not to work. Use `pwd` frequently to check your current directory.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Listing Information on Files

__ 1. Log in to the system. Change to the **myscripts** directory. Display a long listing of the files in the **myscripts** directory. Notice the owner and permissions for the files that you copied in the previous exercise.

Record the permissions for **mycat**. _____

Record the permissions for **mycal**. _____

» Login: **teamxx**

» teamxx's Password: **teamxx**

» $ **cd myscripts**

» $ **ls -l**

__ 2. Now, do a long list on the original **cat** and **cal** files in the /**usr**/**bin** directory and compare the permissions to those in the **myscripts** directory. You own the copies but not the originals.

» $ **ls -l /usr/bin/cat /usr/bin/cal**

__ 3. Change the modification time of **mycal** and **mycat** in the **myscripts** directory. Check to see that the time actually changed. What is another use for the **touch** command? _____

» $ **touch mycal mycat**

» $ **ls -l**

__ 4. Make it so you can reference the **mycal** file in the **myscripts** directory by the name of **home_mycal** in your home directory. Compare the detailed file information for both files.

Is there any difference? _____

What is the link count? _____

» $ **ln mycal /home/teamxx/home_mycal**

-OR-

» $ **ln mycal ../home_mycal**

» $ **ls -l mycal**

» $ ls -l /home/teamxx/home_mycal

-OR-

» $ ls -l ../home_mycal

__ 5. Change the directory to your home directory. Execute **home_mycal**.

What does the output look like?

_____

Now, change permissions on the **home_mycal** file so that you, the owner of the file, have read only permission. Try running the **mycal** command.

Can you do it? _____

Why or why not? _____

» $ cd

» $ home_mycal

» $ chmod 455 home_mycal
$ ls -l home_mycal

» $ myscripts/mycal

__ 6. Remove **home_mycal**. Did that remove **myscripts/mycal**? _____

Why or why not?

_____

» $ rm home_mycal

» $ ls -l myscripts/mycal

Working with File Permissions

__ 7. Change the directory to the **myscripts** directory. Using symbolic notation of the **chmod** command, remove the read permission on the “other” permission bits from the file **mycat**. Check the new permissions.

» $ cd myscripts

» $ chmod o-r mycat

» $ ls -l mycat

__ 8. Using octal notation, change the permissions on **mycat** so that the “owner” permission bits are set to read-only permission with no permission for anyone else. Check the new permissions.

» $ chmod 400 mycat

» $ ls -l mycat

__ 9. Use the **mycat** command to display the contents of the **.profile** file. Did it work?

What happened?

_____

_____

» $ **mycat ../.profile**

-OR-

» $ **mycat /home/teamxx/.profile**

__ 10. Make your home directory the current directory. Check to see if you are in your home directory.

» $ **cd**

» $ **pwd**

Working with Directory Permissions

__ 11. Alter the permissions on the **myscripts** directory so that you have read-only access to it.

» $ **chmod u-wx myscripts**

-OR-

» $ **chmod u=r myscripts**

-OR-

» $ **chmod 455 myscripts**

__ 12. Use a long list to check that you have set the permissions correctly.

» $ **ls -l /home/teamxx**

-OR-

» $ **ls -ld myscripts**

__ 13. Try getting a simple list of the contents of the directory. Try a long list. Did they work?

Why or why not?

_____

_____

» $ **ls myscripts**

» $ **ls -l myscripts**

__ 14. Try to execute **mycal**. Did it work?

Why or why not?

_____

_____

» $ **myscripts/mycal**

__ 15. Try to remove **mycal**. Did it work?

Why or why not?

_____

_____

» $ **rm myscripts/mycal**

__ 16. Return the permissions of **myscripts** back to its original form of **rwxr-xr-x** and then remove **mycal**.

» $ **chmod 755 myscripts**

» $ **rm myscripts/mycal**

__ 17. As time permits, experiment with other permission combinations. When you are through, make sure to change the permissions back to **rwx** for the owner.

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 1. Log in to the system. Change to the **myscripts** directory. Display a long listing of the files in the **myscripts** directory. Notice the owner and permissions for the files that you copied in the previous exercise.
Record the permissions for **mycat**. _____
Record the permissions for **mycal**. _____

» Answer:
-r-xr-xr-x mycat
-r-xr-xr-x mycal

__ 3. Change the modification time of **mycal** and **mycat** in the **myscripts** directory. Check to see that the time actually changed. What is another use for the **touch** command?

» Answer: **touch** can also be used to create empty (zero length) files.

__ 4. Make it so you can reference the **mycal** file in the **myscripts** directory by the name of **home_mycal** in your home directory. Compare the detailed file information for both files. Is there any difference? What is the link count?

» Answer: Only the name is different. The link count is 2 because there are two names in the directory pointing to the same file.

__ 5. Change the directory to your home directory. Execute **home_mycal**. What does the output look like? _____
Now, change permissions on the **home_mycal** file so that you, the owner of the file, have read only permission. Try running the **mycal** command. Can you do it?

_____
Why or why not? _____

» Answer: The output looks like the output from the **mycal** command. After changing permissions on **home_mycal** to read only you will not be able to execute **mycal** because the two files are linked and any changes made to one will be reflected in the other one as well.

__ 6. Remove **myscripts/home_mycal**. Did that remove **myscripts/mycal?**

_____
Why or why not? _____

» Answer: Removing **home_mycal** simply removes the directory entry in **myscripts** that refers to **home_mycal** and changes the link count from 2 to 1. It does not remove the file itself. Thus, at this point the file is only known by one name, **mycal**.

__ 9. Use the **mycat** command to display the contents of the **.profile** file. Did it work? What happened?

» Answer: No. You should have received the message:
`ksh: mycat: Execute permission denied`
It cannot execute because the **x** permission was removed from the file.

__ 13. Try getting a simple list of the contents of the directory. Try a long list. Did they work? Why or why not?

» Answer: The simple list works. All that is needed is **r** permission on a directory to read the name of the files. The long list does not work and displays the message:

```
myscripts/mycat: no permission

myscripts/mycal: no permission
```

In order to get the information about a file, like ownership and permission, you have to be able to get access to what is in the directory. If you do not have permission to be in the directory, you will not be able to access the files in that directory or access information about the files other than their name. **x** permission is required on a directory to access the directory or be in the directory.

__ 14. Try to execute **mycal**. Did it work? Why or why not?

» Answer: No. You will get the message:
`/usr/bin/ksh: myscripts/mycal: not found`

The system cannot find **mycal** because you do not have **x** permission on the directory. In order to access a file in a directory you have to have permission to be in the directory. Without **x** permission on **myscripts** you cannot be in the **myscripts** directory to get access to the files within that directory.

__ 15. Try to remove **mycal**. Did it work? Why or why not?

» Answer: No. You need both **x** and **w** permission on the directory to remove a file.

Exercise Review/Wrap-up

1. Review symbolic and octal form of the **chmod** command.
2. Ask why **mycat** did not work against **.profile** after it was changed to read only permission.

   Answer: **x** permission is required for a command to execute.
3. Review what the results of **ls -l** and **ls -ld** are.
4. Ask why **mycal** would not execute after changing the directory **myscripts** to read only.

   Answer: **x** permission is required on a directory in order to get access to any files, including commands, in a directory.
5. Ask why you could not remove **mycal** after changing the directory **myscripts** to read only.

   Answer: **x** and **w** permission are required on a directory in order to remove something from a directory.

Exercise 6. vi Editor

(with Hints)

Estimated Time

00:45

What This Exercise Is About

The purpose of this exercise is to give the student the opportunity to create and edit files using the most common UNIX editor, **vi**. A clear understanding of the vi editor is critical to successfully complete the rest of the exercises in this course.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Create a file
- Save and exit a file and exit without saving
- Manipulate a file using various cursor movement keys
- Add, delete, and make changes to text within a file
- Set options to customize the editing session
- Invoke command line editing

Introduction

The **vi** editor is based on software developed by the University of California at Berkeley, California, Computer Science Division. The **vi** editor, pronounced “vee-eye” (short for visual), features commands to create, change, append, or delete files. The following exercises will familiarize you with some of the major features and functions of **vi**.

For your assistance, there is a **vi** Command Summary in the Appendix of the Student Notebook.

Known Hardware/Software Problems

If **vi** comes up in **open (line)** mode, it means that the terminal type is not set correctly. Run the command **echo $TERM** to see what it is set

to. Set appropriately to resolve the problem. You may want to add the correct `$TERM` to the student's **.profile** file or change it permanently using SMIT.

Exercise Instructions with Hints

(with Hints)

Estimated Time

00:30

Creating a File

__ 1. Ensure that you are in your home directory. Create a file in your home directory named **vitest**.

» $ **cd**

» $ **pwd**

» $ **vi vitest**

__ 2. When you open a **vi** file, you are automatically placed in command mode. Press the **i** key (insert) to switch to input (text) mode. You can also press the **a** key (append). Use of **i** or **a** simply determines if typing starts before or after the cursor. There is no indication to tell you that you are in input mode.

Switch from input mode to command mode by pressing the **ESC** key. Press **ESC** a second time. Notice that if you press **ESC** twice, you will get a "beep" from the terminal (some ASCII terminals do not beep). The beep indicates that you are in command mode already. Now press **i** again to put you back in input mode. Continue to the next step.

- **i**
- **ESC**
- **ESC** (did you hear a beep)
- **i**

__ 3. Input the following text *exactly* as it is presented line-by-line. Then key in the alphabet, one character per line. Following will show **a-d** but continue on through **z**. Adding the alphabet is an easy way to fill a couple of screens of information needed for later use.

```
This is a training session about the usage of the vi editor. We need
some more lines to learn the most common commands of the editor. We are
now in the entry mode and we will switch right after this to the
command mode.
a
b
c
```

```
d
...
z
```

__ 4. Return to command mode. Write and quit the file. Notice that as soon as you press the **:** (colon), it appears below the last line of your input area. Once the buffer is empty and the file is closed, you will see a message giving the number of lines and characters in the file.

» **ESC** (puts you in command mode)

» **:wq (<shift-ZZ>** or **:x** is another way to write and quit)

Cursor Movement Keys

__ 5. Open **vitest** using **vi**. Notice the bottom line of the file indicates the name of the file and number of characters.

» $ **vi vitest**

__ 6. Using both the arrow keys and the **h**, **j**, **k**, **l** keys, practice moving the cursor down one line, up one line, right a couple characters, and back a couple characters.

» **j** (down a line)

» **k** (up a line)

» **l** (right a character)

» **h** (left a character)

» Repeat using the appropriate arrow keys.

__ 7. You may not want to cursor one character or one line at a time throughout an entire file. Practice using cursor movement keys to work around by page or by line. Using the cursor movement keys from the previous step, position your cursor at the first line of the file. While in command mode, do the following:

i. Move forward one page.

ii. Move back one page.

iii. Move cursor to last line in the file.

iv. Move cursor to first line in the file.

v. Move cursor to line 4 of the file.

vi. Move cursor to end of line.

vii. Move cursor to beginning of line.

» **<Ctrl-f>** or press the Page Down (**PgDn**) key. (There is no Page Down key on ASCII terminals and the key may not do this function on your terminal.)

» **<Ctrl-b>** or press the Page Up (**PgUp**) key. (There is no Page Up key on ASCII terminals and the key may not do this function on your terminal.)

» **<Ctrl-u>**

» **<shift-G>**

» **1<shift-G>** or **:1 Enter**

» **4<shift-G>** or **:4 Enter**

» **$**

» **0** (this is a zero)

__ 8. Move your cursor to the top of the file. Search for the word `entry`. Your cursor should be on the `e`. Switch to input mode and add the word "**text**". Do not forget the space after the word.

» **1<shift-G>** or **:1**

» **/entry**

» **i**

» **text**

__ 9. Move the cursor to the space after the word `mode` on the same line. Insert a comma. Remember, you are still in input mode.

» **ESC**

» Position the cursor to the space after mode

» **i,** (comma)

__ 10. Enter command mode. Position the cursor anywhere on the line beginning with "`some more lines`". Insert a blank line to form two paragraphs.

» **ESC**

» Position cursor on line starting "`some more lines`"

» **o** (lower case **o** opens the line after the cursor)

__ 11. Opening a blank line as in the previous step, automatically puts you in input mode; therefore, return to command mode. Now save the changes you have made so far, but DO NOT exit the editor.

» **ESC**

» **:w**

__ 12. While still in command mode, remove the alphabetic characters **c**, **e**, **g** but leave the blank lines in their place; in other words, do not delete the entire line, just the character. Then go back and remove the blank lines. This will give you practice using two of the delete functions.

» Position cursor on **c**; Press **x**

» Position cursor on **e**; Press **x**

» Position cursor on **g**; Press **x**

» Position cursor on each of the blank lines; Press **dd**

__ 13. Now replace the alphabetic character **h** with a **z**.

» Position the cursor on the **h**

» Press **r** (for replace)

» **z**

__ 14. You just decided you really do not want to save the changes to the alphabetic characters. Quit the editing session without saving the changes made since the last save.

» **:q!**

__ 15. Edit **vitest** one more time. First, copy the first paragraph (including the blank line) one line at a time to the end of the file. When that is complete, copy the second paragraph all at once to the end of the file.

» $ **vi vitest**

» Position cursor on line one; Press **yy**

» **<shift-G>**; Press **p**

» **2<shift-G>**; Press **yy**

» **<shift-G>**; Press **p**

» **3<shift-G>**; Press **yy**

» **<shift-G>**; Press **p**

» **4<shift-G>**; Press **2yy**

» **<shift-G>**; Press **p**

__ 16. You just decided that the lines you just added to the end of file do not look right. Delete them all with one command.

» Position the cursor on the first copied line at the bottom of the file to be deleted

» Count the number of lines to delete

» **5dd** (your number may be different if you moved the blank line as well)

__ 17. Now, before you do anything else with this file, you decide you need to imbed the current date and time as the first line of the file. Do this without leaving the **vi** editor.

» **:!date > datefile**

» Do not press enter when you see the message `Press return to continue`

» **:0r datefile**

» Press **Enter** twice to continue

Using Set to Customize the Editing Session

__ 18. Options can be set temporarily in an editing session using the **set** command. Go back to the top of your file. Ensure you are in command mode and set the following commands:

a. Set automatic word wrap 15 spaces before the right margin.

b. Display the INPUT MODE message when in input mode.

c. Turn line numbering on

» **1<shift-G>**

» **ESC**

» **:set wrapmargin=15** (no spaces around the **=**)

» **:set showmode**

» **:set number**

__ 19. Test each of the options set in the previous instruction.

» Lines should be numbered.

» Enter input mode using **i** or **a**. You should see an INPUT MODE message at the bottom right of your display.

» Key in a couple lines of miscellaneous text to test automatic word wrap.

» Enter command mode by pressing **ESC**. The INPUT MODE message should have disappeared from your display.

__ 20. Write the file and quit the editor.

» **:wq**

Command Line Editing

__ 21. Now that you are familiar with **vi** modes and commands, practice command line editing. To set up your session to use command line editing, use the **set -o vi** command.

» $ **set -o vi**

__ 22. Now you can recall previously executed commands, edit them, and resubmit them. Let's build a command history to work with. List (simple, not long) the contents of the directory /**usr**. Display the contents of the file /**etc/filesystems**. Echo `hello`.

» $ **ls /usr**

» $ **cat /etc/filesystems**

» $ **echo hello**

__ 23. Suppose you want to edit one of the commands you just executed. Press the **ESC** key to get to **vi** command mode. Try pressing the **k** key several times to go up the list of commands. Try **j** to go down. This recall of commands is essentially looking through a buffer of commands that you previously executed. The commands are actually stored in your **.sh_history** file in your home directory.

» **ESC**

» **k** (go up list of commands in buffer)

» **j** (go down list of commands in buffer)

__ 24. Retrieve the **ls** command. Use the **l** key to move your cursor to the **/** in **/usr**. (Note: the arrow keys tend to wipe your line out. You have to use the **l** key for right and **h** for left.) Use the **i** key to insert text and change this command to be a long list. Execute it.

» **k** (to the **ls /usr** command)

» **l** (to get to the **/**)

» **i** (to get into input mode. You could have used **a** to append if the cursor was on the space before the **/**)

» **-l**

» **Enter**

__ 25. Recall the **cat** command. This time list the contents of the /**etc**/**passwd** file.

» **ESC**

» **k** (to get to the previous **cat** command)

» **l** (to move the cursor to the f in filesystems)

» **D** (to erase rest of line, or **dw** to erase the word)

» **a** (to append text)

» **passwd**

» **Enter**

__ 26. Recall the **cat** command. Go to the end of the line (remember **$**). Add to the end of the command to pipe the output to **wc** to count just the lines.

» **ESC**
» **k** (to get to the last **cat** command)
» $
» **a**
» **| wc -l**
» **Enter**

END OF EXERCISE

Exercise Review/Wrap-up

1. You are in a **vi** editing session but not sure what mode you are currently in. What should you do?

   Answer: Press ESC to get to command mode. Then enter a command for editing or a command to get into input mode.

2. How can you visibly tell if you are in command or input mode?

   Answer: By setting **set showmode** you will see the message INPUT MODE at the bottom right of the screen when in input mode. No message at the bottom right of the screen, indicates you are in command mode.

3. Does it matter how many times you concurrently press ESC while in **vi**?

   Answer: No, but repetitive pressing of ESC may cause the keyboard to beep.

4. You entered the **set** commands from the keyboard while in a **vi** session. Using info, read about creating a file named **.exrc** in your home directory to contain these commands so they are automatically set up for you each time you invoke **vi**. You may want to try this in class when you have some extra time. I will be glad to assist you.

5. How many of you mastered the command line editing feature using **set -o vi**? If you are not an expert typist, this feature will come in very handy during some of the upcoming exercises which will contain many long commands that will be executed more than once.

Exercise 7. Shell Basics

(with Hints)

Estimated Time

00:45

What This Exercise Is About

This exercise will familiarize the students with basic shell operations.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Use wildcards for file name expansion
- Redirect standard in, standard out, and standard error
- Use pipes to provide the output of one process as input to another process
- Perform command grouping and line continuation

Introduction

Understanding the use and manipulation of the shell is considered a foundation for understanding AIX user interfaces. You will use commands to experiment with the shell features discussed in the “Shell Basics” lecture.

Exercise Instructions with Hints

(with Hints)

Estimated Time

00:30

Wildcards

__ 1. Type **cd** to get back to your home directory. (Your home directory is the one you use when you log in.)

» $ **cd**

» $ **pwd**

__ 2. Execute a simple **ls** to list the non-hidden files in your home directory. Now use the **ls** command with a wildcard character to list these files. What is the difference in output of these two commands?

_____

Why?

_____

» $ **ls**

» $ **ls ***

__ 3. Change to the /**usr**/**bin** directory. List just those files starting with the letter **a**.

» $ **cd /usr/bin**

» $ **ls a***

__ 4. List all two character file names.

» $ **ls ??**

__ 5. List all file names starting with the letters **a**, **b**, **c**, or **d**.

» $ **ls [abcd]***

-OR-

» $ **ls [a-d]***

__ 6. List all files except those beginning with **c** through **t**. This will be a long list. You might want to pipe the output to **pg** or **more**. Did you get any file names that you did not expect? _____ If so, do you know why?

» $ **ls [!c-t]* | pg**

__ 7. Return to your home directory.

» $ **cd**

Redirection

__ 8. Using the **cat** command and redirection, create a file called **junk** containing a few lines of text. Use **<Ctrl-d>** at the beginning of a new line when you have finished entering text and want to return the shell $ prompt. List the file contents to verify your update.

» $ **cat > junk**

Type in several lines of junk for your file
<Ctrl-d> on a new line to return to the shell prompt

» $ **cat junk**

__ 9. Append more lines of text to the file you have created using the **cat** command and redirection. List the file contents to verify your update.

» $ **cat >> junk** (no spaces between the >>)

» $ **cat junk**

__ 10. Mail the file **junk** to yourself. Wait a minute and open your mail, delete it, and quit the program.

» $ **mail teamxx < junk**

» $ **mail**

» **? t**

» **? d**

» **? q**

Pipes, Tees, and Filters

__ 11. Using the **ls** command, list the files in your current directory. Make a note of the number of files:

» $ **ls**

__ 12. List the files in your current directory, but this time redirect the output to the file **temp**.

» $ **ls > temp**

__ 13. Use the appropriate command to count the number of words in the **temp** file. Is this the same count as in instruction 11? _____ If not, why not?

_____

Display the contents of **temp**. Remove the file.

» $ **wc -w temp**

» $ **cat temp**

» $ **rm temp**

__ 14. This time use a pipe to count the number of files in your current directory. Was the result what you expected this time? _____ Is it the same as in instruction 11?

_____

» $ **ls | wc -w**

__ 15. Use the command you created in instruction 14, but this time insert a **tee** in the middle trapping the result of the list in a file called **junk2**. Did you get the number displayed on the screen? _____

Check the contents of **junk2** to make sure that it contains what you expected.

» $ **ls | tee junk2 | wc -w**

» $ **cat junk2**

__ 16. List in reverse order the contents of your current directory. Send the results of the reverse listing to a file named **junk3**, and to a program to count the number of words in the reverse listing. Append the final count to **junk3**. Remember to use the append version of redirection. In this particular case, you may get unexpected results if you do not. It might not be a straight overwrite because the file is being used twice in the same command. Experiment if you are curious.

» $ **ls -r | tee junk3 | wc -w >> junk3**

» $ **cat junk3**

__ 17. There is a special file in the /**dev** directory that represents your terminal. Display the file name associated with your terminal. Output will be something like `tty0`, `lft0`, or `pts/x`. Repeat the command from instruction 16 with two exceptions:

1) Rather than using **junk3**, tee the output to the special file that represents your terminal (/**dev/<your_terminal_name>**).
2) Do not append the results of the wc command to **junk3**. Have the count display to your terminal.

» $ **who am i**

» $ **ls -r | tee /dev/lft0 | wc -w**

Command Grouping and Line Continuation

__ 18. On the same command line, display the date, who is logged in, the name of your current directory, and the names of the files in your current directory. Do these commands have any relationship to each other?

» $ **date ; who ; pwd ; ls**

__ 19. The primary purpose of this exercise instruction is to use line continuation with a command that is too long to fit on one command line. The secondary purpose is to test what you have learned so far by letting you create an incredibly long command string.

You can choose to break the line anywhere you feel comfortable, but do not type past the right edge of the screen. When completed, test your output by displaying the contents of the files that were created. This should be one long command connected by pipes and redirection.

1) Do a long listing of the files in your home directory including hidden files.
2) Capture the output to a file named **reverse.listing** and send the same output to a program that will count only the number of words.
3) Capture the number of words and place the number in 4 files named **file1** through **file4**.
4) Finally, send the output to a program to count the number of lines captured in the previous instruction and redirect that number to a file named **file5**.

» $ **ls -al | tee reverse.listing | wc -w | tee file1 \<Enter>**
> | tee file2 | tee file3 | tee file4 | wc -l > file5
(the **>** symbol at the start of the second line is the secondary prompt; you do not type this)

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 2. Execute a simple **ls** to list the non-hidden files in your home directory. Now use the **ls** command with a wildcard character to list these files. What is the difference in output of these two commands?

Why? Answer: With **ls *** you got the contents of any subdirectories in your home directory because the shell expanded the * before the **ls** command executed. The shell expanded it to be the names of all the files (remember, directories are files) in the directory. When you ask **ls** to list an ordinary file, it does, but when you ask it to list a directory, it lists the *contents* of the directory, not the directory name itself. Use the **-d** option of the **ls** command to have it list the directory itself instead of the contents, if you want to see information on the directory only.

__ 6. List all files except those beginning with **c** through **t**. This will be a long list. You might want to pipe the output to **pg** or **more**. Did you get any file names that you did not expect? _____ If so, do you know why?

Answer: When you asked for all files except those beginning with **c-t**, you will only execute lowercase entries. Uppercase entries like **R** and **M** will be displayed.

__ 13. Use the appropriate command to count the number of words in the temp file. Is this the same count as in instruction 11? If not, why not?

Display the contents of **temp**. Remove the file.

Answer: The results of the **wc -w** command should be one greater than the original count. The shell sets up the command line prior to executing the command; thus, the **temp** file was created as an empty file prior to the execution of **ls** allowing **temp** to be included in the count.

__ 14. This time use a pipe to count the number of files in your current directory. Was the result what you expected this time? _____ Is it the same as in instruction 11? _____

Answer: Yes. No **temp** file was created to capture the output for redirection.

__ 15. Use the command you created in instruction 14, but this time insert a **tee** in the middle trapping the result of the list in a file called **junk2**. Did you get the number displayed on the screen?

Check the contents of **junk2** to make sure that it contains what you expected.

Answer: The number incremented by one because of the addition of the **junk2** file.

Hint: There may be times when you may not get the results you thought. These are independent processes communicating with each other. If the **ls** command finishes processing before the shell has a chance to create the **junk2** file, then **junk2** would not be included in the count.

__ 18. On the same command line, display the date, who is logged in, the name of your current directory, and the names of the files in your current directory. Do these commands have any relationship to each other? _____

Answer: Command grouping is just a shortcut for executing non-related commands from the same command line. They have NO relationship to each other.

Exercise Review/Wrap-up

1. Is there any difference in `ls *` and `ls -a`?

   Answer: Yes, the * does not display hidden files and **-a** does not display what is in subdirectories.

2. In instruction 8, how were you using the `cat` command?

   Answer: As an editor.

3. How do you exit `cat` when you are using it as an editor?

   Answer: `<Ctrl-d>` on a line by itself. This is similar to ending the `mail` program.

4. In instruction 17, you found the file name associated with your terminal by issuing the command `who am i`. What other command could you have used?

   Answer: `who`. As a hint, you can execute the command `tty` to also see the file name associated with your terminal.

5. How many of you keyed in the long command in instruction 19 correctly the first time? What symbol is used to represent the secondary prompt?

   Answer: "**>**" This can get confusing when using redirection with line continuation as it is the same symbol used for redirecting standard out.

Exercise 8. Using Shell Variables

(with Hints)

Estimated Time

00:45

What This Exercise Is About

The student will define and utilize variable and command substitution to set the shell environment and utilize quoting to override the shell interpretation of metacharacters.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- List shell built-in variables
- Set up variable substitution to define or alter the environment
- Use command substitution to set variables equal to the output of a command
- Use the three methods of quoting to allow metacharacters to be used literally instead of interpreted

Introduction

This exercise contains three sections: variable substitution, command substitution, and quoting. Knowledge of the first two sections, variable and command substitution, is required to perform the third section, quoting.

Caution: Throughout this exercise, the single quotes and the back quotes look very similar. The single quotes look like this `'`, and the back quotes like this `` ` ``. The back quote may look different on the keyboard than it does as printed in this exercise.

Common Student Problems

When students begin the Command Substitution section, it is a common mistake for the students to use the single quote rather than

the back quote. If students are having trouble executing command substitution, ensure they are using back quotes. In this exercise the single quotes appear as ', and the back quotes appear as `. To save some confusion for those that are doing the Hints section, draw the forward and backward appearance of the single quotes versus the back quotes on the board. The students can then use this as reference during the exercises. The students are warned about this in the Introduction section. It is worth repeating for those students who did not take the time to read the Introduction.

Exercise Instructions with Hints

(with Hints)

Estimated Time

00:30

Variable Substitution

__ 1. Display the shell built-in variables.

```
» $ set
```

__ 2. Set a variable named `lunch` to `pizza` and a variable named `dinner` to `ham`. Display the value of the variables using **`echo`**. Locate them in the list of variables.

```
» $ lunch=pizza
» $ dinner=ham
» $ echo $lunch ; echo $dinner
» $ set
```

__ 3. Using the variables you just defined, display the message, `Lunch today is pizza and dinner is ham.`

```
» $ echo Lunch today is $lunch and dinner is $dinner
```

__ 4. Using the variables you just defined, display the message, `Lunch today is hamburgers`.

```
» $ echo Lunch today is ${dinner}burgers
```

__ 5. Remove the value of both variables. Check to be sure they are no longer included in your list of variables.

```
» $ unset lunch
» $ unset dinner
» $ set
```

__ 6. Display the value of your primary and secondary prompt strings.

```
» $ echo $PS1
» $ echo $PS2
```

__ 7. Change the primary prompt string to `"You Rang?"`. (Single quotes will also work) Why is it necessary to use the quotes with `"You Rang?"`?

```
» $ PS1="You Rang?"
```

__ 8. Change your secondary prompt string to "What Else?". Test it with the **ls** command using line continuation. End the command. Reset both prompt strings back to their original values. Why are quotes needed around the > when resetting the PS2 variable? _____

» You Rang? **PS2="What Else?"**

» You Rang? **ls -l **
What Else?

» **<Ctrl-c>**

» You Rang? **PS1="$ "**

» $ **PS2="> "**

__ 9. Check the value of the variable related to your home directory. Reset that variable to change your home directory to /**bin**. Use the **cd** and **pwd** commands to test the effects of this change.

» $ **echo $HOME** (You could have checked the value using *set*)

» $ **HOME=/bin**

» $ **cd ; pwd**

__ 10. Log out and log back in. What is your home directory? _____ Why?

_____

Note: If you are working in an **aixterm** session, after keying **exit**, press the right mouse button and select New Window to get back to an **aixterm** session.

» $ **exit**

» login: **teamxx**

» teamxx's Password:

» $ **cd ; pwd**

» $ **echo $HOME**

Command Substitution

__ 11. Display your list of variables. Reissue the command but send the output to the **wc** command to get the number of variables that are currently set.

» $ **set**

» $ **set | wc -l**

__ 12. Using command substitution, **echo** the following:
There are # variables currently set
where # is the number of variables.

» $ **echo There are `set | wc -l` variables set**

OR

- `$ echo There are $(set | wc -l) variables set`

__ 13. Each user ID configured on the system is represented by one line in the **/etc/passwd** file. Applying your knowledge of command substitution, echo a message that displays:
`There are # users created on the system`
where # is the number of line entries in **/etc/passwd**.

» `$ echo There are 'cat /etc/passwd | wc -l ' users created on the system`

OR

» `$ echo There are $(cat /etc/passwd | wc -l) users created on the system`

Quoting

__ 14. Using all three methods of quoting, **banner** the literal symbol `*`. Why do all three work?

_____

» `$ banner '*'`

» `$ banner "*"`

» `$ banner \*`

__ 15. Ensure you are in your home directory. Create a directory in your home directory named **quoting**.

» `$ cd`

» `$ pwd`

» `$ mkdir quoting`

__ 16. Change to the **quoting** directory. Create a zero-length file in the **quoting** directory named **filea**. Create a variable named `n` set to the value of `hello`. Test what you have done by displaying the contents of **quoting** and the value of `n`.

» `$ cd quoting`

» `$ touch filea`

» `$ n=hello`

» `$ ls`

» `$ echo $n`

__ 17. From the **quoting** directory, execute the following five commands. Record the output. Check the *Solutions* section for the expected output.

i. $ **echo '* $n `ls` $(ls)'**

_____

ii. $ **echo "* $n `ls` $(ls)"**

_____

iii. $ **echo * \$n \`ls\` \$\(ls\)**

_____

iv. $ **echo * $n `ls` $(ls)**

_____

v. $ **echo * $n ls**

_____

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 7. Change the primary prompt string to `"You Rang?"`. (Single quotes will also work) Why is it necessary to use the quotes with `"You Rang?"` ?

Answer: Double quotes must be used because of the space between the words.

__ 8. Change your secondary prompt string to `"What Else?"`. Test it with the **`ls`** command using line continuation. End the command. Reset both prompt strings back to their original values. Why are quotes needed around the `>` when resetting the `PS2` variable? _____

Answer: By using quotes around the >, the symbol will not be interpreted as redirection.

__ 10. Log out and log back in. What is your home directory? _____ Why?

Answer: Your home directory should be back to the default /**home**/**teamxx**. Changing variables from the command line only sets the value for the length of the login session. Once you log out, the variable is removed from your environment.

__ 14. Using all three methods of quoting, **`banner`** the literal symbol `*`. Why do all three work?

Answer: All three work because the shell always negates wildcards no matter what method of quoting is used.

__ 17. From the **quoting** directory, execute the following five commands. Record the output. Check the *Solutions* section for the expected output.

Answers:

`* $n \`ls\` $(ls)`
Single quotes suppresses everything between them.

`* hello filea filea`
Double quotes do command and variable substitution only.

`* $n \`ls\` $(ls)`
Backslash negates the character following it. Note the use of a backslash in front of each back quote.

`filea hello filea filea`

`filea hello ls`

Exercise Review/Wrap-up

1. It is recommended that any user-defined variables be named using lowercase characters. Why?

   Answer: Built-in variables are defined in uppercase. By using lowercase characters for user-defined variables, you will more easily differentiate between the two types. This makes it easier to change or remove user-defined values.

2. What does quoting mean?

   Answer: Escaping the meaning of metacharacters, variable substitution, and command substitution from the shell. It is called quoting because two of the three methods use single or double quotes.

3. If a metacharacter is quoted or escaped from the shell, what will process that metacharacter and any related string (for which we otherwise might have carried out variable substitution, command substitution, or file name substitution)?

   Answer: The metacharacters and any effected strings are passed as parameters to the command being executed by the shell. The first word on the command prompt line which is not a variable assignment is treated by the shell as a command to be executed.

Exercise 9. Controlling Processes

(with Hints)

Estimated Time

00:45

What This Exercise Is About

This exercise familiarizes the student with process manipulation and process control.

What You Should Be Able to Do

After completing this exercise students should be able to:

- Monitor processes by using the **ps** or **jobs** command
- Control processes by using the **kill** or **jobs** command
- Display current process ID

Introduction

In this exercise you will use commands to experiment with process control to get a better understanding of your process environment. You will identify the processes associated with your terminal session, work with variables in parent and child processes and terminate processes you have started.

Common Student Problems

Sometimes the command that is running in the background finishes before the students complete the next step which assumes that the background process is still running. Simply have the students kick off the background process again and try to get them to do the next step a little bit quicker.

Since many students are still struggling with **vi**, the simple scripts they are asked to create might be a bottleneck for the exercise. If you hear a lot of beeping coming from a student's terminal, see if you can help them with **vi** so they do not lose sight of the purpose of the exercise.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Structure

__ 1. Log in to the system and display your current process ID(PID).

» $ **echo $$**

__ 2. Create a subshell by entering `ksh`. What is the process ID of the subshell?

Is it different from your login process?

» $ **ksh**

» $ **echo $$**

__ 3. Enter the command **`ls -lR / > outfile 2> errfile &`** and then execute the command which displays all of your running processes. The **`ls`** command will terminate when it finishes listing all the files in the directory tree.

» $ **ls -lR / > outfile 2> errfile &**

» $ **ps -f**

__ 4. Terminate your child shell. What happens if you type `exit` from your login shell?

» $ **exit**

Process Environment

__ 5. Display all your variables that are in your current process environment.

» $ **set**

__ 6. Create a variable `x` and set its value to `10`. Check the value of the variable. Again, display all your current variables.

» $ **x=10**

» $ **echo $x**

» $ **set**

__ 7. Create a subshell with **`ksh`**. Check to see what value variable `x` holds in the subshell. What is the value of `x`? _____ List the subshell current variables. Do you see a listing for `x`? _____

» $ **ksh**

» $ **echo $x**

» $ **set**

__ 8. Return to your parent process. Set the value of variable `x` so that its value will be inherited by your child processes. Verify this by creating a subshell and checking on the value of variable `x`.

» $ **exit**

» $ **export x=10**

» $ **ksh**

» $ **echo $x**

__ 9. Change the value of `x` to `200` in the subshell. Check that the value was changed.

» $ **x=200**

» $ **echo $x**

__ 10. Go back to the parent process. Check on the value of `x` in this environment. Was the change in the subshell exported back to the parent?

» $ **exit**

» $ **echo $x**

__ 11. Create a shell script and name it **sc1**. It should read:

```
pwd; cd /; pwd
```

» $ **vi sc1**

```
pwd
cd /
pwd
```

» Press the `ESC` key followed by `:wq` to save the file.

__ 12. Make the file **sc1** executable and run the program. What directory are you in now?

Why?

» $ **chmod 700 sc1**

» $ **sc1**

» $ **pwd**

__ 13. Create another shell script and name it **sc2**. Have it read:

```
var1=hello; var2=$LOGNAME; export var1 var2
```

» $ **vi sc2**
var1=hello
var2=$LOGNAME
export var1 var2

» Press the ESC key followed by :wq to save the file.

__ 14. Make **sc2** executable and run the program. When it is finished, examine the values of the variables var1 and var2. What values do var1 and var2 have?

Why?

» $ **chmod 700 sc2**

» $ **sc2**

» $ **echo $var1 $var2**

__ 15. Run the **sc2** program again, this time by forcing it to run in the current shell. When it is finished, check the values for var1 and var2**.** What values do var1 and var2 have now?

Why?

» $ **. sc2**

» $ **echo $var1 $var2**

Job Control

__ 16. Create a shell script and name it **sc3**. It should read:

```
sleep 120
ls -lR / > outfile 2> errfile &
```

Make it executable. Start the script with the command:

$ **./sc3 > outfile 2> errfile**

in the foreground.

» $ **vi sc3**
(press i to insert text)
sleep 120
ls -lR / > outfile 2> errfile &
(press the ESC key followed by :wq to save the file)

» **chmod 700 sc3**

» $ **./sc3 > outfile 2> errfile**

__ 17. Suspend the job you just started.

» $ **<Ctrl-z>**

__ 18. List all the jobs that you are running on the system and restart the above job in the background.

» $ **jobs**

» $ **bg %jobno**

__ 19. Bring the job back to the foreground.

» $ **fg %jobno**

__ 20. Once the command finishes executing, restart it again in the background, display the process ID, and log off.

» $ **./sc3 > outfile 2> errfile &**

» $ **jobs -l**

» $ **exit** (you will get a message that says you have jobs running)

» $ **exit**

__ 21. Log in. Check to see if the process is still running.

» Login: **teamxx**

» teamxx's Password: **teamxx**

» $ **ps -ef**

__ 22. Start the **sc3** script with the **nohup** command, reference it using an explicit path and put it in the background. Do not forget to redirect the output from **sc3,** note its process ID and job number and then log off.

» $ **vi sc3**
(press i to insert text)
sleep 120
ls -lR / > outfile 2> errfile &
(press the ESC key followed by :wq to save the file)

» **chmod 700 sc3**

» $ **nohup ./sc3 > sc3.out 2> sc3err &**

» $ **jobs -l**

» $ **exit** (you will get a message that says you have jobs running)

» $ **exit**

__ 23. Log in. Check to see if the process is still running. Hint: search for its process ID.

» Login: **teamxx**

» teamxx's Password: **teamxx**

» $ **ps -ef**

__ 24. When the process is complete, display the file that contains your output. (Hint: if you did not specify an output file, **nohup** will send the output to **nohup.out**.)

» $ **pg /home/teamxx/outfile**

__ 25. Rerun the **sc3** script you just created placing it into the background but not using the **nohup** command. Note its process id and job number. Apply the **nohup** to the process ID of the background process **sc3** and then log off.

Log back into the system and verify that the process is still running.

» $ **./sc3 > sc3.out 2> sc3err &**

» $ **jobs -l** (note the PID of the job you just started in the background)

» $ **nohup -p <PID>**

» $ **exit** (you will get a message that says you have jobs running)

» $ **exit**

» Login: **teamxx**

» teamxx's Password: **teamxx**

» $ **ps -ef**

Terminating a Process

__ 26. Use the **ls -lR /** command we have been using to start a long running job in the background. Note the process ID that is provided when you begin the background process. _____

» $ **ls -lR / > outfile 2> errfile &**

__ 27. If you did not record the process ID when you first started the command in the background, how would you find it? _____

Once you know the process ID, kill the process. Check to be sure it was killed.

» $ **ps -f**

» $ **kill <pid>**

» $ **ps -f**

__ 28. Repeat instruction 26 above. Kill the process using the job number rather than the process ID. Check to be sure the job was killed.

» $ **ls -lR / > outfile 2> errfile &**

» $ **jobs**

» $ **kill %jobno**

» $ **jobs**

-OR-

» $ **ps -f**

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 2. Create a subshell by entering `ksh`. What is the process ID of the subshell?

Is it different from your login process?

Answer: Your child process ID will always be different from the parent and is unique on the system.

__ 4. Terminate your child shell. What happens if you type `exit` from your login shell?

Answer: You will log off the system.

__ 7. Create a subshell with **`ksh`**. Check to see what value variable `x` holds in the subshell. What is the value of `x`? _____ List the subshell current variables. Do you see a listing for `x`? _____

Answer: The value of `x` is null. In the output from the **`set`** command, `x` is not shown.

__ 10. Go back to the parent process. Check on the value of `x` in this environment. Was the change in the subshell exported back to the parent?

Answer: No because the subshell runs in a different process than the parent.

__ 12. Make the file **sc1** executable and run the program. What directory are you in now?

Why?

Answer: Your original directory because the **`cd`** command executed in a subshell. When the child terminates, the parent resumes with its original environment.

__ 14. Make **sc2** executable and run the program. When it is finished, examine the values of the variables `var1` and `var2`. What values do `var1` and `var2` have?

Why?

Answer: The values of both `var1` and `var2` are null. The **`sc2`** script runs in a subshell. When it completes, control is returned to the parent process. Variables set in child processes are not available to parent processes.

__ 15. Run the **`sc2`** program again, this time by forcing it to run in the current shell. When it is finished, check the values for `var1` and `var2`. What values do `var1` and `var2` have now?

Why?

Answer: `var1` is hello and `var2` is your logname. By starting the **`sc2`** script with the **`.`** you are forcing it to run in the current process. Therefore, a new process is not spawned and the variable is set and stays in the current process' environment.

__ 26. If you did not record the process ID when you first started the command in the background, how would you find it? _____

Answer: Use **`ps -f`**.

Exercise Review/Wrap-up

1. Review the differences between a foreground and background process and how to start each.
2. Ask what happened to the values of `var1` and `var2` when the script was started normally.

   Answer: The value of `var1` and `var2` were not passed back to the parent shell.
3. Ask what happened to the values of `var1` and `var2` when the script was started with a `.`

   Answer: When a script is started with a `.` it is executed in the current shell and thus no subshell is created. This functions like setting variables within the current shell at the command line.
4. Ask how they might go about suspending a job they started in the background.

   Answer: To suspend a job started in the background, bring it into the foreground and then use **`<Ctrl-z>`** to suspend it.

Exercise 10. Customizing the User Environment

(with Hints)

Estimated Time

00:30

What This Exercise Is About

When users log in, they generally prefer their environment to be customized to meet their specific needs. In this exercise, the student will customize their environment with some very useful functions that are invoked every time they log in.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Customize **.profile** and **.kshrc** files
- Set alias definitions

Introduction

Half way through the exercise, you will change your primary prompt from a `$` to the name of your current directory. This changed prompt string will be reflected from that point on in the exercises. This will look different from what you are use to seeing in previous exercises.

If you are working in an X Windows session, when instructed to log out, execute one of the following commands: **`$ su -`** or **`$ login`**.

Common Student Problems

The most common student problem with this exercise is the lack of **`vi`** experience. Help students with all **`vi`** problems allowing them to concentrate on the concepts of customizing their environment rather than mastering **`vi`**.

If students working in an X Windows session are having problems logging out, ensure they are using one of the two suggested commands as explained in the *Introduction*.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Customizing .profile and .kshrc

__ 1. To customize your environment and have it take effect every time you log in, you must incorporate the changes in a file that is read at login. Ensure you are in your home directory. Edit your **.profile** file to add the following functions:

1) Change the primary prompt string to reflect the current directory.
2) Display a message at login which contains your login name and the time you logged in.
3) Define an alias named **dir** that invokes the **ls -l** command.
4) Automatically set up the command line editing facility.

» $ **cd**

» $ **pwd**

» $ **vi .profile**
PS1='$PWD => ' (no space between = and >)
echo User $LOGNAME logged in at $(date)
alias dir='ls -l'
set -o vi
:wq

__ 2. Test your customization by re-executing your **.profile**. You can choose to log out and back in, or simply rerun it using the dot notation. Once you have done that, execute and answer the following:

1) Did your message display? _____
2) Is your prompt the name of your home directory? _____
3) Change to the /**etc** directory. Did your prompt change? _____
4) Using **dir** do you get a long listing of your current directory? _____
5) Invoke **dir** using command line editing.

If you answered NO to any question, edit your **.profile** and fix it.

» $ **logout**

» Login: **teamxx**

» teamxx Password: **teamxx**

-OR-

» `$ . .profile`

__ 3. Once you have your customized **.profile** setup and functioning, open a subshell.

Answer the following questions:

i. Is your prompt the name of the current directory? _____

ii. Does the value of the alias dir still work? _____

iii. Can you invoke command line editing? _____

» **/home/teamxx=>** `ksh`

__ 4. Exit from the subshell and return to your home directory. Most settings, with the exception of system variables, only apply to the current environment and are not passed to subshells (child processes). To pass **alias** or **set** customized settings down to subshells, the ENV variable must be set in your **.profile** file along with the existence of a customized **.kshrc** file.

Revise your **.profile** and create the appropriate **.kshrc** file to support the **alias** and **set** customization you did in instruction 1.

In the .**profile** file, remove the '`alias dir`' and '`set -o vi`' customizations. Add the ENV variable assignment. Export both PS1 and ENV.

Add the **alias** and **set** customizations (you just removed from **.profile**) to **.kshrc**.

» **<Ctrl-d>**

» **/home/teamxx=>** `cd`

» **/home/teamxx=>** `vi .profile`

```
PS1='$PWD=> '
ENV=/home/teamxx/.kshrc
export PATH PS1 ENV
echo User $LOGNAME logged in at $(date)
```

(or whatever worked for you)

```
:wq
```

» **/home/teamxx=>** `vi .kshrc`

```
set -o vi
alias dir='ls -l'
:wq
```

__ 5. Test your customization by re-executing your **.profile** file. Open a subshell and answer the following questions:

i. Is your prompt the name of the current directory? _____

ii. Is the value of the alias dir still working? _____

iii. Can you invoke command line editing? _____

» $ **. .profile**
/home/teamxx=> ksh

__ 6. Exit the subshell and return to your login shell. Display a listing of all currently set alias names and locate the **dir** alias.

» **/home/teamxx=><Ctrl-d>**

» **/home/teamxx=> cd**

» **/home/teamxx=> alias**

__ 7. Temporarily unalias **dir** without editing the **.kshrc** file. Then display the list of alias settings again and ensure that it is no longer defined. Try executing **dir**.

» **/home/teamxx=> unalias dir**

» **/home/teamxx=> alias**

» **/home/teamxx=> dir**

__ 8. The **dir** alias is still in your **.kshrc** file but is not set. The **unalias** command removed it from the list of current alias names. Invoke **.kshrc** to automatically add **dir** back in the alias list. Execute **dir**.

» **/home/teamxx=> . .kshrc**

-OR-

» Log out and log back in to reactivate **.kshrc**

» **/home/teamxx=> dir**

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 2. Test your customization by re-executing your **.profile**. You can choose to log out and back in, or simply rerun it using the dot notation. Once you have done that, execute and answer the following:

1) Did your message display? _____
2) Is your prompt the name of your home directory? _____
3) Change to the /**etc** directory. Did your prompt change? _____
4) Using **dir** do you get a long listing of your current directory? _____
5) Invoke **dir** using command line editing.

If you answered NO to any question, edit your **.profile** and fix it.

Answers:

1) A message similar to the following should be displayed:
   User teamxx logged in at Thu Apr 19 14:34:26 CST 2001
2) Your primary prompt string should be similar to:
   /home/teamxx=>
3) **/home/teamxx=> cd /etc**
4) **/etc=> dir**
   You should see a long listing of the current directory.
5) **ESC**
   k
   /etc=> dir

__ 3. Once you have your customized **.profile** setup and functioning, open a subshell.

Answer the following questions:

i. Is your prompt the name of the current directory? _____
ii. Does the value of the alias dir still work? _____
iii. Can you invoke command line editing? _____

The answer to all three questions should have been NO and output that looks similar to the following:

a) $ (Unless you thought ahead and exported PS1 as well in **.profile**)
b) $ **dir**
   ksh: dir: not found.
c) $ **ESC k** (you should see a square bracket and letter **k**)

__ 5. Test your customization by re-executing your **.profile** file. Open a subshell and answer the following questions:

i. Is your prompt the name of the current directory? _____

ii. Is the value of the alias dir still working? _____

iii. Can you invoke command line editing? _____

The answers should be *YES* in all cases since the `PS1` variable was exported in **.profile** making it available to subshells. The `ENV` variable was added to **.profile** and exported allowing **.kshrc** and its contents to be executed and passed to all subshells. You should see something similar to the following:

i. Your primary prompt string should be similar to:
/home/teamxx=>

ii. **/home/teamxx=>** **`dir`**
You should see a long listing of the current directory.

`iii.ESC`
`k`
`/home/teamxx=> dir`

Exercise Review/Wrap-up

1. What is the difference between **.profile** and **.kshrc?**

   Answer: The **.profile** file is only read at login; whereas, the **.kshrc** file is read at login and each time a child process is created.

2. Do you have to have a **.kshrc** file when using the Korn shell?

   Answer: No. You can put everything in the **.profile** file, but you may see unexpected results, as you experienced during the exercises.

3. In the file **.kshrc**, you added the command **`set -o vi`**. There is another way to ensure that you automatically have this command activated. In your **.profile** file add and export the variable **`EDITOR=/usr/bin/vi; export EDITOR`**. This not only sets up **`set -o vi`**, but it will allow **`vi`** to be the default editor for some programs that require a default editor to be defined like **`mail`**.

Exercise 11. AIX Utilities (1)

(with Hints)

Estimated Time

00:30

What This Exercise Is About

The purpose of this exercise is to become familiar with some of the most helpful data tools available with AIX.

What You Should Be Able to Do

After completing this exercise, students should be able to execute recursive searches on directories for files that meet specific criteria.

Introduction

This exercise is designed to give you experience using the **`find`** command.

Using the command line editing feature will be very helpful during this exercise as some of the commands can get quite lengthy and will be repeated in many instructions.

This exercise shows the `$` prompt; however, unless you reset the `PS1` variable from the prior exercise, you will see your current directory as your prompt.

Common Student Problems

Many students will use command line editing as many of the commands in this exercise are long and used many times. Students who did not master the use of **`vi`** will struggle when attempting to use command line editing. Assist any student who needs help with **`set -o vi`** allowing them to concentrate on the concepts of the exercise.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

The `find` Command

__ 1. Find and display all the files in the /**tmp** directory.

» $ **`find /tmp`**

__ 2. Find all files in your home directory that begin with the letter *s* and have **`ls -l`** automatically execute on each file name found as a result of the search operation.

» $ **`find . -name 's*' -exec ls -l {} \;`**

OR

» $ **`find . -name "s*" -ls`**

__ 3. Repeat the search in the previous step, but interactively prompt the user to display the long list on each file.

» $ **`find . -name 's*' -ok ls -l {} \;`**

__ 4. Find all files starting from the /**usr** directory that are owned by the userid **uucp**. Modify the command line to count the number of files owned by **uucp**. There may be some directories that you do not have permission to read. This will cause a permission denied message to be displayed. Redirect all error messages to a file called **errfile**.

» $ **`find /usr -user uucp 2> errfile | wc -l`**

__ 5. Display the file **errfile** from the previous instruction to see if any errors messages were encountered.

» $ **`pg errfile`**

__ 6. To demonstrate that **`find`** recursively searches all directories and subdirectories from the search path down, do the following:

a. Ensure you are in your home directory.

b. Make a subdirectory called **level1**.

c. Create a zero-length file named **letter1** in the subdirectory **level1.**

d. Change to the **level1** subdirectory.

e. Make a subdirectory under **level1** called **level2**.

f. Create a zero-length file named **letter2** in the subdirectory **level2**.

g. Change to your home directory.

h. From your home directory issue the command to list all files starting with the letter `l`. Record the names displayed.

i. From your home directory issue the command to **`find`** only files starting with the letter `l`. Record the names displayed.

```
» $ cd
» $ mkdir level1
» $ touch level1/letter1
» $ cd level1
» $ mkdir level2
» $ touch level2/letter2
» $ cd
» $ ls l*
» $ find . -name 'l*' -type f
```

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 6. To demonstrate that **find** recursively searches all directories and subdirectories from the search path down, do the following:

a. Ensure you are in your home directory.

b. Make a subdirectory called **level1**.

c. Create a zero-length file named **letter1** in the subdirectory **level1**.

d. Change to the **level1** subdirectory.

e. Make a subdirectory under **level1** called **level2**.

f. Create a zero-length file named **letter2** in the subdirectory **level2**.

g. Change to your home directory.

h. From your home directory issue the command to list all files starting with the letter `l`. Record the names displayed.

i. From your home directory issue the command to **find** only files starting with the letter `l`. Record the names displayed.

Answer: The output is predictable. **find** goes to the end of the tree and will display the path names of files meeting the selection criteria. In our example, we used a relative path name as the starting directory for our **find**. As a result, **find's** output is presented as relative pathnames as well.

Exercise Review/Wrap-up

- When using find why is quoting used when wildcards are part of the search criteria?

  Answer: The single quotes, double quotes, and backslash character can all protect the metacharacter from being interpreted by the shell. The interpretation must be done by **`find`**. Use of any of the three protection methods should produce the same results.

Exercise 12. AIX Utilities (2)

(with Hints)

Estimated Time

00:45

What This Exercise Is About

The purpose of this exercise is to become familiar with some of the most helpful data tools available with AIX.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Search text files for pattern matching
- Extract specific fields within a file
- Sort lines in a file
- Display the first or last few lines of a file.
- Log in to a remote system
- Transfer files between systems
- Save and restore files using the **`tar`** command

Introduction

This exercise is designed to give you experience using some AIX data tools.

Common Student Problems

Many students will use command line editing since many of the commands in this exercise are long and used many times. Students who did not master the use of **`vi`** will struggle when attempting to use command line editing. Assist any students who needs help with **`set -o vi`** allowing them to concentrate on the concepts of the exercise.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

The `grep` Command

__ 1. Find all lines in the /**etc**/**passwd** file for user names that start with **team**.

» $ **grep team /etc/passwd**

__ 2. Find all lines in the /**etc**/**passwd** file that begin with the letter `t`.

» $ **grep '^t' /etc/passwd**

__ 3. Find all lines in /**etc**/**passwd** that contain a digit `0-9`.

» $ **grep '[0-9]' /etc/passwd**

__ 4. Repeat the search in the previous instruction, but this time display only the number of lines that contain the pattern.

» $ **grep -c '[0-9]' /etc/passwd**

__ 5. Use the **ps** and **grep** commands to display the processes initiated by users other than yourself, and pipe the output to the **more** command.

» $ **ps -ef | grep -v teamxx | more**
where **teamxx** is your login name.

The `sort` Command

__ 6. Display the content of the /**etc**/**passwd** file in alphabetic order. Next, display the contents of the file in reverse order.

» $ **sort /etc/passwd**

» $ **sort -r /etc/passwd**

The `head` and `tail` Command

__ 7. Display the first 10 lines of /**etc**/**passwd**.

» $ **head /etc/passwd**

__ 8. Display the first 5 lines of /**etc**/**passwd**.

» $ **head -5 /etc/passwd**

__ 9. Display the last 10 lines of /**etc**/**passwd**.

» $ **tail /etc/passwd**

__ 10. The **tail** command is also handy for stripping out header information from the output of a command. First, list all processes currently running on your system. Notice the headings. Next, display all processes running on your system excluding the header information.

```
» $ ps -ef | more
» $ ps -ef | tail +2 | more
```

The tn, ftp and tar Commands

__ 11. Log in to any remote system in your classroom. If you are not sure about the name of the remote system ask your instructor. Use one of the **teamxx** user IDs that have been supplied by your instructor.

```
» $ tn sysx
```

__ 12. Execute the **hostname** command and verify that you really work on the remote system.

```
» $ hostname
```

__ 13. Change to the /**tmp** directory and create a new file **testfile1**.

```
» $ cd /tmp
» $ vi testfile1
```

__ 14. Log out from the remote system.

```
» $ exit
```

__ 15. On your local system create a new directory **remote_files**.

```
» $ mkdir remote_files
```

__ 16. Transfer the remote file /**tmp**/**testfile1** to your local system. The file should be stored in the subdirectory **remote_files**.

```
» $ ftp sysx
  ftp> get /tmp/testfile1 remote_files/testfile1
  ftp> quit
```

__ 17. Verify that the file has been copied to your local system.

```
» $ cd remote_files
» $ ls
```

__ 18. Stay in the **remote_dir** subdirectory and use the **tar** command to save all files in this directory. Create an archive file /**tmp**/**archive.tar** and save all files relatively.

```
» $ tar -cvf /tmp/archive.tar *
```

__ 19. Verify the content of the archive file.

```
» $ tar -tvf /tmp/archive.tar
```

__ 20. Restore all files from your archive into the /**tmp**-directory.

» $ **cd /tmp**

» $ **tar -xvf archive.tar**

END OF EXERCISE

Exercise Review/Wrap-up

- There are three methods of quoting: single quotes, double quotes, and backslash. Why is it recommended to use single quotes with **grep**?

  Answer: The shell is permitted to look inside double quotes for variable and command substitution. If double quotes are used around the metacharacter $ with **grep**, the shell will grab the $ and try to perform variable substitution. Use of single quotes allows the $ to be passed to **grep** instead. **grep** has a list of its own metacharacters. The $ to grep means end of line, not variable substitution.

Exercise 13. AIX Utilities (3)

(with Hints)

Estimated Time

00:30

What This Exercise Is About

This exercise allows you to experiment with additional helpful utilities that can be used in the AIX environment.

What You Should Be Able to Do

After completing this exercise, students should be able to use **`find`**, **`xargs`**, and **`file`** to manipulate files.

Introduction

You will be manipulating ordinary files and directories using commands discussed in lecture. Where there is more than one way to invoke a command, you will see an -OR- between the possible solutions in the **Exercise Instructions with Hints** section.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Using `find`, `xargs`, *and* `file`

__ 1. Verify that you are in your `$HOME` directory. Create a subdirectory called **newdir**. Change to this directory and create five empty files in this directory with the names of **1**, **2**, **3**, **4**, and **5**.

» $ **pwd**

» $ **mkdir newdir**

» $ **cd newdir**

» $ **touch 1 2 3 4 5**

__ 2. Show a simple list of the contents of the **newdir** directory. Now, list the contents of the **newdir** directory and pass the output to **xargs** to copy the files and rename them with the prefix **file** so the resulting copied file's name is **file1**, and so forth. Verify that the files were copied and the names assigned accordingly.

» $ **ls**

» $ **ls | xargs -t -I {} mv {} file{}**

» $ **ls**

__ 3. Using **find**, **xargs**, and **grep**, display the names of the files under your home directory which contain the string `AIX`.

» $ **cd**

» $ **pwd**

» $ **find . -type f | xargs grep AIX**

__ 4. Find out in which directory the **find** command is located. Determine the type of file (executable, ASCII, dir, etc) of the **find** command.

» $ **which find**

-OR-

» $ **whereis find**

-OR-

» $ **whence find**

» $ **file /usr/bin/find**

-OR-

» $ **file `whence find`** (note the back quotes)

__ 5. Using the **find** command to recursively list the file under your home directory, determine the type of each file. This may be accomplished in two ways. You might first creating a file, named **myfiles** and then determine the file types of the files listed in **myfiles**. Alternatively you might do the entire task in a pipe, thus eliminating the need to create the file, **myfiles**.

» $ **find $HOME > myfiles; file -f myfiles | pg**

-OR-

» $ **find $HOME | xargs file | pg**

END OF EXERCISE

Exercise 14. AIX Utilities (4)

(with Hints)

Estimated Time

00:35

What This Exercise Is About

This exercise allows you to experiment with additional helpful utilities that can be used in the AIX environment.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Use **`diff`**, **`cmp`**, and **`dircmp`** to compare files and directories
- Use **`compress`**, **`zcat`**, and **`uncompress`**
- Use **`cat`** to display non-printable characters

Introduction

You will be manipulating ordinary files and directories using commands discussed in lecture. Where there is more than one way to invoke a command, you will see an -OR- between the possible solutions in the **Exercise Instructions with Hints** section.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Using `diff`, `cmp`, `dircmp`

__ 1. Create a file called **list1**. In **list1**, list the names of several people you know, one line per name. Copy **list1** to a file called **list2**. Edit **list2** and make the following changes:

- Change the spelling of one of the names.
- Remove one of the names.
- Add a new name.

» $ **`vi list1`**
(Add several names)

» $ **`cp list1 list2`**

» $ **`vi list2`**
(Make the changes listed in a-c above)

__ 2. Using **`diff`**, compare the contents of **list1** and **list2**.

» $ **`diff list1 list2`**

__ 3. Using **`cmp`**, compare the contents of **list1** and **list2**. Then invoke a complete or long comparison of the contents of both files.

» $ **`cmp list1 list2`**

» $ **`cmp -l list1 list2`**

__ 4. Using **`dircmp -d`**, compare your home directory with the home directory of another user account on your system (**teamyy**).

» $ **`dircmp -d /home/teamxx /home/teamyy | pg`**

Using `compress`, `uncompress`, `zcat`

__ 5. Copy the file /**etc**/**magic** to a file in your home directory named **mymagic**. Do a long listing on **mymagic** and record the number of bytes in the file: _____

» $ **`cp /etc/magic mymagic`**

» $ **`ls -l mymagic`**

__ 6. Using the verbose option with **`compress`**, compress **mymagic**. Record the percentage of compression, _____, and the name of the compressed

file, _____. Do a long listing on the file and record the number of bytes. _____ Compare the number to the number in the previous instruction.

» $ **compress -v mymagic**

» $ **ls -l mymagic.Z**

__ 7. Using **zcat**, expand and view the contents of **mymagic.Z**. You may want to page it as it is a large file.

» $ **zcat mymagic.Z | pg**

__ 8. Using **uncompress**, restore the compressed file back to its original file. Invoke a long listing and record the number of bytes. _____ The number should be the same as the number in Step 5.

» $ **uncompress mymagic.Z**

» $ **ls -l mymagic**

Displaying Non-Printable Characters

__ 9. In your home directory, create a file named **invis** and type a few lines that include random tabs, spaces, **Ctrl-G**'s, and so forth, between the words. Display the file.

» $ **vi invis**

» $ **cat invis**

__ 10. Notice in the instruction above that when you displayed the contents of **invis** it did not look quite right. Display and locate all the non-printable characters to determine where you used spaces, tabs, control characters, and so forth.

» $ **cat -vte invis**

__ 11. Create a directory named **invisdir** but insert an accidental <**Ctrl-g**> somewhere in the name.

» $ **mkdir invisdir^G**

__ 12. Invoke the following four commands. When asked to key in the **invisdir** name, do *NOT* enter the <**Ctrl-g**> you originally included as part of the name.

1) Invoke a listing of files and directories in your home directory (**invisdir** should be included as part of the output).
2) Try to invoke a long listing on the **invisdir** directory.
3) Try to remove the **invisdir** directory.
4) Repeat instruction a. above to see if there are any non-printable characters in the **invisdir** directory name that made instructions 2 and 3 fail.

» $ **ls**

» $ **ls -ld invisdir** (This should fail)

» $ **rmdir invisdir** (This should fail)

» $ **ls | cat -vt**

__ 13. Using a method of your choice, successfully remove the **invisdir** directory.

» $ **rmdir invisdir^G** (include the <**Ctrl-g**> in the name where it appears in your listing)

-OR-

» $ **mv invisdir^G invisdir**

» $ **rmdir invisdir**

-OR-

» $ **ls -i**

» $ **find . -inum <i-node #> | xargs rmdir**

END OF EXERCISE

Exercise Review/Wrap-up

What was your percentage of compression for the **mymagic** file? Was anyone's compression percentage different?

Exercise 15. Additional Shell Features

(with Hints)

Estimated Time

01:00

What This Exercise Is About

After you have been using AIX for a while, you will find certain characteristics of your environment that you would like to customize along with some tasks that you execute regularly that you would like to automate.

This exercise will introduce you to some of the more common constructs used to help you write shell scripts in order to customize and automate your computing environment.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- List common constructs used in writing shell scripts
- Create and execute simple shell scripts

Introduction

You need not have any programming experience to perform this exercise. Refer to the unit in the Student Notebook for help with the syntax of constructs when creating the shell scripts in this exercise.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Writing shell scripts

__ 1. Create a shell script named **parameters** that will echo the five lines that follow using predefined special variables set by the shell to fill in the blanks. Execute the script using the positional parameters **10 100 1000**.

```
The name of this shell script is ________.
The first parameter passed is number ____.
The second parameter passed is number ____.
The third parameter pass is number ____.
Altogether there were _____ parameters passed.
```

» $ **vi parameters**

```
echo The name of this shell script is $0.
echo The first parameter passed is number $1.
echo The second parameter passed is number $2.
echo The third parameter passed is number $3.
echo Altogether there were $# parameters passed.
```

» $ **chmod +x parameters**

» $ **parameters 10 100 1000**

__ 2. Using conditional execution, create a shell script named **checkfile** that will check to see if the file named **parameters** exists in your directory, and if it does, use a command to show the contents of the file. Execute the script.

» $ **vi checkfile**

```
ls parameters && cat parameters
```

» $ **chmod +x checkfile**

» $ **checkfile**

__ 3. Modify the **checkfile** script and change the name of the file from **parameters** to **noname** (check to ensure that you do NOT have a file by this name in your current directory). Also, using conditional execution, if the **ls** command was NOT successful, display the error message, `The file was not found.` Execute the script. What else got displayed?

```
» $ vi checkfile
  ls noname && cat noname || echo The file was not found
» $ checkfile
```

__ 4. Modify the **checkfile** script so that error messages from the **ls** command do not appear on the screen. Execute the script.

```
» $ vi checkfile
  ls noname 2> /dev/null && cat noname || echo The file was not
  found
» $ checkfile
```

__ 5. Modify the **checkfile** script to accept a single parameter from the command line as input to the **ls** and **cat** commands. Execute the script twice, once using the file named **parameters** and again using the file named **noname**.

```
» $ vi checkfile
  ls $1 2> /dev/null && cat $1 || echo The file was not found
» $ checkfile parameters
» $ checkfile noname
```

Using for, test, *and* if

__ 6. Using the **for** loop, modify the **checkfile** script to accept multiple files as input from the command line instead of just one. If the files are found, display all of them. If the files are not found, display the error showing all file names that were not found. Look in your directory and jot down a few valid file names that you can use as input. Execute the script using valid and invalid file names.

```
» $ vi checkfile
  for x in $*
  do
    ls $x 2> /dev/null && cat $x || echo $x was not found
  done
» $ ls
» $ checkfile filename filename filename
  (Where filename is replaced by valid and invalid file names from your
  directory)
```

__ 7. Change the **checkfile** script to use an **if** statement and **test** command rather than conditional execution to check if the **ls** command was successful. Execute the script as you did in the previous step. (Hint: Return codes play a part in this script.)

```
» $ vi checkfile
  for x in $*
  do
    ls $x 2> /dev/null
```

```
  if [[ $? -eq 0 ]]
    then cat $x
    else echo $x was not found
  fi
done
```

```
» $ checkfile filename filename filename
```

Using **while** *and* **expr**

__ 8. Create an endless **while** loop that will echo `Out to Lunch` every 5 seconds in a script named **lunch**. Execute the script. When you have seen enough, break the loop.

```
» $ vi lunch
while true
do
  echo Out to Lunch
  sleep 5
done
```

```
» $ chmod +x lunch
```

```
» $ lunch
<Ctrl-c>
```

__ 9. From the command line, display the results of multiplying 5 and 6.

```
» $ expr 5 \* 6
```

__ 10. Now using **expr**, create a shell script named **math** to multiply any two numbers when entered as input from the command line. Execute the script multiplying 5 times 6. Experiment with any other two numbers.

```
» $ vi math
expr $1 \* $2
```

```
» $ chmod +x math
```

```
» $ math 5 6
```

END OF EXERCISE

Solutions

Following are the solutions for those instructions that include questions:

__ 3. Modify the **checkfile** script and change the name of the file from **parameters** to **noname** (check to ensure that you do NOT have a file by this name in your current directory). Also, using conditional execution, if the **ls** command was NOT successful, display the error message, `The file was not found.` Execute the script. What else got displayed?

Answer: An error message from the **ls** command.

Exercise Review/Wrap-up

1. In instruction 1, is the **`echo`** command necessary as part of the file?

   Answer: Yes. It is the vehicle used to display the text that follows to standard out.

2. In instruction 8, is the use of the **`while`** loop an efficient method of letting anyone in your office know you are out to lunch?

   Answer: No, as this uses unnecessary CPU cycles every 5 seconds to display the message. As was discussed in Exercise 1. Using the System, use of **`echo`** and **`banner`** to display messages that you are out of the office is not good for security reasons.

Exercise 16. Using AIXwindows

(with Hints)

Estimated Time

01:00

What This Exercise Is About

This exercise provides an opportunity to use AIXwindows.

What You Should Be Able to Do

At the end of the lab, students should be able to:

- Start AIXwindows
- Manipulate screen windows using AIXwindows
- Open a new **aixterm** window
- Customize motif application on launch (optional)
- (optional) Use the **xhost** command and DISPLAY environment variable to execute an X Client on a remote system

Introduction

It will be necessary to perform this machine exercise through a VNC session. Be sure to check with your instructor if you have any questions regarding the terminal you should use.

While in the AIXwindows environment, you may wish to minimize or close any windows not needed to prevent the terminal screen from becoming too cluttered.

This exercise also includes an optional exercise. Verify with your instructor that the machine setup will support the optional exercise.

Common Student Problems

This exercise includes optional steps 23-28, which use the **xhost** command and the DISPLAY variable to run X Clients remotely. If the classroom environment is using separate AIX systems or LPARs for each student, they all must be attached to a network and configured to

support TCP/IP. In the optional exercise, students will execute the **hostname** command to learn the TCP/IP name of their system. Each TCP/IP name in the classroom must be unique.

The instructor may wish to test the TCP/IP network prior to running this machine exercise to verify connectivity. This can be accomplished by the following:

- Type **hostname** at one AIX system.
- Type **hostname** at another AIX system.
- At either system, execute the **ping** command to check for connectivity to the remote system. For example, **ping sys2**.
- Use **<Ctrl-d>** to end the **ping**. The **ping** statistics should show zero packets lost. If the statistics indicate lost packets, check the network cabling as well as the TCP/IP setup of the systems.

You may get an error message when starting a motif application from the command line:

_X11TransTRANS (ibmSHMConnect) () can't connect: errno = 68

We did not determine the cause of this message (it may be related to the VNC environment), but it does not seem to result in any problems and can be safely ignored.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Starting AIXWindows

__ 1. Start your terminal emulator application, connect to the AIX system, and log in.

» login: **teamxx**

» password **<your password>**

__ 2. Before we can start the VNC server application, you will need to set a VNC password for your remote session. Run the **vncpasswd** program, and when prompted, enter a password. Set your password to the same as your username (for example, if your userid was **team01**, set the password to **team01**).

» **vncpasswd**

__ 3. Start the VNC server application. Make note of the hostname/IP address and session number for the VNC server that is started.

Hostname: _____

IP Address: _____

Session number: _____

»**$ vncserver**

```
New 'X' desktop is <hostname>:<session number>

Starting applications specified in
/home/team01/.vnc/xstartup
Log file is /home/team01/.vnc/<hostname>:<session
number>.log
```

» **$ host <hostname>**

__ 4. Switch back to your lab workstation or lab portal facility and launch a VNC viewer. Enter in the **IPaddress:session** where appropriate, and the password. If all is correct, a window should appear with the AIXWindows environment in it, running under your assigned userid.

» Start a VNC viewer on your lab workstation (this will be either your classroom workstation or a lab portal facility, depending on the lab configuration), and specify the hostname or IP address and session number. When asked,

supply the password you set in Step 3. The AIXWindows environment should appear within the VNC client window.

Working with Windows

__ 5. Verify that the **aixterm** is the active window.

» Move the mouse pointer to the **aixterm** and click with the left mouse button. If it is not already the active window, the window frame will change color to indicate that this is now the active window.

__ 6. Using the **aixterm**, try typing some AIX commands such as **ls**, **date**, **cal**, and **whoami**.

__ 7. Resize the width of the window.

» Move the mouse pointer to the window frame on the right edge of the **aixterm** window. Note that the pointer shape changes. Press the left mouse button down and keeping the button pressed, move the mouse.

» An outline should appear showing you the new size of the window. A feedback window should also appear indicating the size of the new window in rows and columns.

__ 8. Change the height and width of the window, simultaneously.

» Move the pointer to one of the window frame corners. Then resize the window with the mouse as above.

__ 9. Drag the **aixterm** from one side of the screen to the other.

» To move a window on the screen, move the mouse pointer to the title area of the window.

» Using the left mouse button, press and hold while moving the mouse. An outline of the window should appear indicating the new position of the window. Release the left mouse button.

» The window is now at a new location.

» A feedback window shows you information about window position. That information can be used with the *-geometry* keyword used to position windows initially.

__ 10. Use the options in the window menu to move and resize the window.

» Click with the left mouse button on the window menu button (the button to the left of the title area). A menu should appear.

» Select the Size and Move options and move the mouse. They should perform the same functions as above. Alternatively, you can use the keyboard cursor control keys instead of moving the mouse.

» Notice that with the Size option, the edge or corner that is moved to resize the window is the first of them that the mouse pointer comes into contact with.

» Click the left mouse button or press enter when you have finished resizing.

__ 11. View the window menu again. Why do you think some items may be greyed out?

__ 12. Open the window menu on the **aixterm**, but now type the letter **m** rather than clicking **move**. Note that this is another way to move a window.

» Click the window menu button with the left mouse button.

» The window menu will appear.

» Notice that the menu items have letters underlined (for example, the **m** in **move**).

» Press **m** on the keyboard. These functions are not case sensitive. What happens?

» These defined keys are known as mnemonics.Try the mnemonics for some of the other functions.

__ 13. The window menu also contains key sequence definitions (for example **Alt+F7**). These key bindings are known as accelerators.

What happens when you try pressing the **Alt+F7** key when the menu is posted?

What happens if you try a mnemonic when the menu is not posted?

__ 14. Iconify (minimize) the **aixterm** window. Once it is an icon, restore it back to the screen.

» Use the left mouse button to click the small square button just to the right of the title area. This should turn the window into an icon.

» To restore the window, click the icon with the left mouse button. (When working in VNC, you ma need to scroll the VNC window to locate the icon.) The window menu is again displayed, with certain options activated. Using the left mouse button, click **restore**. This will restore the window with its previous location and size. Double-clicking an icon will also restore that window.

__ 15. Maximize the **aixterm** window. What happens? Once it is maximized, resize the window to a smaller size.

» Using the left mouse button, click the large square button to the right of the title area. The **aixterm** window should fill the screen.

» Make the window smaller using any of the techniques you have learned previously.

Using the root Window

__ 16. Use the root menu to open another **aixterm** window.

» Move the mouse pointer to the grey or root window that fills the screen.

» Click the right mouse button. The root menu will be displayed.

» From the root menu, point to the option **New Window**. When the right mouse button is released, a new **aixterm** or **dtterm** will be displayed.

__ 17. Start another **xclock** from the root menu.

» Use the right mouse button to click in the root window. The root menu will be displayed.

» Note that **Clients** shows an arrow indicating that a sub-menu will be displayed.

» While holding down the right mouse button, move the mouse pointer to the **Clients** option. A sub-menu will be displayed.

» Move the mouse to the **Clock** option and release the right mouse button.

Cut and Paste Functions

__ 18. Within a window, use the vi editor to create a file called **tempfile**. Add a few lines of text to this file, but do not exit.

» **vi tempfile**

» Add a few lines of text

__ 19. Within a second **aixterm** window, use the **vi** editor to create another file called **tempfile.new**. Go into insert mode but do not add any text to this file at this time.

» **vi tempfile.new**

» **i** to access insert mode

__ 20. Copy a few lines of text from **tempfile** in the first window to **tempfile.new** in the second window. When you have completed this step, exit **vi** in both windows.

» To cut text from **tempfile**, move the mouse pointer to the beginning of the line you wish to copy. Press the left mouse button and drag the pointer to where you wish the selection to end. The selected text should be in reverse video.

» When the left mouse button is released, the text will be placed into a buffer.

» Move the mouse pointer to **tempfile.new** and left click to bring it to the foreground. Then press the shift key and click the right mouse button. The text will be placed beginning at the cursor location. (On a locally attached graphics terminal and 3-button mouse, the middle button would normally be used here)

» <Esc> and `:q!`

__ 21. You have now completed this machine exercise. You may either try the optional steps that follow, end AIXwindows or lock your terminal.

» Continue with the optional steps.

-OR-

» Move the mouse pointer to the root menu and press the right mouse button.

» On the root menu choose **Clients**.

» From the **Clients** menu choose **Screen Lock**.

» To unlock the screen, attempt some action (such as clicking the mouse or pressing enter) and, in response to the prompt, type your AIX user password.

__ 22. When you are done with working on this AIXWindows exercise, whether at this point or somewhere in the optional steps), be sure to go to the last step (step 28) and terminate the vncserver on your system.

*Command Line Options for aixterm (optional)_*

__ 23. The **aixterm** command has many command line options. View these options using the **aixterm -help** command. You will need to pipe the output to **pg** or **more** as there is a lot of information.

» **aixterm -help | pg**

If a printer is available, you may wish to print this information.

» **aixterm -help | qprt**

__ 24. Start an **aixterm** from the command line. Give the window the following characteristics:

background color	lightskyblue
foreground color	forestgreen
font	rom10.iso1
title	My Window

full cursor

scrollbar

» **aixterm -bg lightskyblue -fg forestgreen -fn rom10.iso1 -T "My Window" -fullcursor -sb &**

Why do you think this window is smaller than the others?

__ 25. Start an **xclock** from the command line within one of the windows. Give the clock the following characteristics:

background color	white
foreground color	red
hands on the dial	blue
second hand	update every second

» **xclock -bg white -fg red -hd blue -update 1 &**

The Client-Server Model (optional)

__ 26. Use the **xhost** command to enable all other clients to access your X server.

» **xhost +**

You should see the message "Access control disabled, clients can connect from any host".

__ 27. Have another user (if possible, on a different server) try and start an **aixterm** and display it to your AIXWindows session. They will need to change the value of their DISPLAY variable to the hostname and X server number (which is the same as the VNC session number).

» **export DISPLAY=<hostname>:X.0** (Where **X** is the VNC session number)

»**aixterm &**

__ 28. In the new window that appeared, use the **id** command to verify that the window was started from the other user. Check the value of the DISPLAY variable. It should indicate the name of your host.

» **id** (should show other user)

» **hostname** (should show the other teams host, if different)

» **echo $DISPLAY** (should show your system name)

__ 29. From the remote system's window, execute the **xcalc &** command. From which system is the calculator being executed? You can verify this with the **ps** command. When you have completed this step, close the remote system's window.

» **xcalc &**

» **ps**

» **exit**

__ 30. You have now completed the optional machine exercise. Shut down the VNC AIXWindows session from your original terminal emulator session.

» **vncserver -kill :X** (Where **X** is the VNC session number)

END OF EXERCISE

Solutions

__ 11. View the window menu again. Why do you think some items may be greyed out?

Answer: The “greyed out” options are not active for this window.

__ 13. The window menu also contains key sequence definitions (for example **Alt+F7**). These key bindings are known as accelerators. What happens when you try pressing the **Alt+F7** key when the menu is posted? What happens if you try a mnemonic when the menu is not posted?

Answer: The *Alt+F7* key combination is yet another way to move a window. Note that the other window menu options also have accelerators.

The accelerators can be used even if the window menu is not an option. The mnemonics can only be used if the window menu is open.

__ 23. Start an **aixterm** from the command line. Give the window the following characteristics: ...

Why do you think this window is smaller than the others?

Answer: This window is smaller than the others because of the font. The window is still 80 characters wide by 25 characters high, but appears smaller due to a small font. To list all available fonts use **xlsfonts** command.

__ 28. From the remote system's window, execute the **xcalc &** command. From which system is the calculator being executed? You can verify this with the **ps** command. When you have completed this step, close the remote system's window.

Answer: The **xcalc** command should be executing on the remote system, but displayed on your system.

Exercise 17. Using the Common Desktop Environment (CDE)

(with Hints)

Estimated Time

01:00

What This Exercise Is About

This exercise introduces you to the features of CDE.

What You Should Be Able to Do

At the end of the lab, students should be able to:

- Recognize the various CDE controls on the Front Panel
- Use the Help Manager
- Start both an **aixterm** and **dtterm** terminal window
- Use the File Manager to navigate the directory structure, create new files (using the CDE text editor) and directories (folders), and place a file icon in the workspace backdrop
- Optionally, use the Calendar control to view the calendar, set appointments, and create reminders

Introduction

This exercise is designed to provide an introduction to the features of CDE. You will use the Help Manager to obtain information as needed. Much of your work in this exercise will be with the File Manager, which is one of the most useful CDE functions.

If time permits, an optional exercise is included on the CDE calendar functions. Feel free to explore the other functions of CDE. In the next unit and exercise, you will learn to customize your CDE environment.

Instructor Note

In the next two exercises, it will be necessary for the students to access CDE. The instructor will need to set the students' systems up so that CDE will be supported. It is suggested that this is performed before or after class so as not to take up class time performing system administration functions.

If the students are accessing AIX from a graphics terminal (LFT), it will be necessary to perform the following steps on each system in the classroom:

- **`# smit dtconfig`**
- Press **`F4`** on the SMIT `Select System User Interface` screen.
- On the list, move the cursor to `AIX CDE 1.0` and press **`Enter`**.
- Press `enter` on the SMIT `Select System User Interface` menu.
- Verify the output to make sure it ran successfully. Then, press **`F10`**.
- Reboot the system: **`# shutdown -Fr`**.

Once the systems are rebooted, the students will see the CDE login manager window, where they can log in to the CDE desktop.

If the AIX system is being accessed via a VNC client, then the students will set up their own VNC server session to start CDE. The steps are included in the lab exercise.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Exploring the Front Panel

__ 1. Be sure that you have terminated the vncserver that was started for the Using AIXwindows exercise. The instructions for this were in the last step of that exercise

__ 2. If you are logging in on a locally connected graphics terminal, then enter your user information at the **Login Manager** panel.

If you are using VNC to create the CDE session, then connect to the AIX system using the ASCII terminal application and log in with your userid. Rename the existing **xstartup** file in the **$HOME/.vnc** directory to xstartup.bak.

» Connect and log in as your userid.

```
»$ cd .vnc
»$ mv xstartup xstartup.bak
```

__ 3. Ensure that your present working directory is your home directory. Start the VNC server application. Make note of the hostname/IP address and session number for the VNC server that is started.

Hostname: _____

IP Address: _____

Session number: _____

```
»$ cd
»$ vncserver

New 'X' desktop is <hostname>:<session number>

Starting applications specified in
/home/team01/.vnc/xstartup
Log file is /home/team01/.vnc/<hostname>:<session
number>.log
```

» **$ host <hostname>**

__ 4. Switch back to your lab workstation or lab portal facility and launch a VNC viewer. Enter in the **IPaddress:session** where appropriate, and the password. If all is

correct, a window should appear with the AIXWindows environment in it, running under your assigned userid.

» Start a VNC viewer on your lab workstation (this will be either your classroom workstation or a lab portal facility, depending on the lab configuration), and specify the hostname or IP address and session number. When asked, supply the password you set in Step 3. The AIXWindows environment should appear within the VNC client window.

__ 5. The CDE environment, by default, launches the Application Manager and File Manager. Close those two windows (we will restart them later).

» In the upper left corner of each window click on the file menu and then click on the click on Close at the bottom of that menu.

__ 6. Locate the CDE Front Panel; you may need to scroll the desktop window to find it at the bottom of the window. Find the following components of the CDE Front Panel (do not click on them, just locate them):

Workspace Switch Buttons

» The four push buttons in the middle of the panel.

Style Manager

» The icon with the mouse and color palette.

File Manager

» The icon that looks like a file cabinet drawer next to the calendar.

Application Manager

» The icon that looks like a file cabinet drawer with a pencil (next to help).

Personal Application Manager

» The icon that looks like a piece of paper and a pencil.

Clock, Calendar

» Note the time and date.

Mail

» The envelope icon.

Trash Can

» Icon on the right end of the panel.

Exit Icon

» The small icon to the right of the Workspace Switch buttons.

Move Handles

» Left and right ends of the panel.

Menu and Iconify Buttons

» The menu button is in the top left corner of the panel.

» The iconify button is in the top right corner.

__ 7. Move the Front Panel to the top of the screen.

» Click the Move Handle at the right or left end, hold the button down and drag the panel by moving the mouse.

__ 8. Iconify the Front Panel and then restore it.

» To iconify, click the top right corner of the panel.

» To restore the Front Panel: click the icon and select **Restore.**

Work with the Help Manager

__ 9. From the Help subpanel note how options exist so that you can access AIX online documentation. The infocenter menu item will not work unless the infocenter facility has been configured on the lab system and has connectivity to an infocenter server. When you have reviewed the various Help functions, close the Help windows.

» Click the arrow above the Help Manger icon and examine the resulting subpanel. Use the File menu Close item to close each window. When finished, click on the arrow above the Help Manager icon a second time in order to close the menu.

Starting a Terminal Window

__ 10. Start an **aixterm** Terminal Window.

» Select the Application Manager control (the file cabinet icon next to Help).

» Double-click the **Desktop Tools** icon with the left button of the mouse to display the tools available. One of them is **aixterm** .

» Double-click the **aixterm** icon with the left mouse button.

Now you have a terminal window where commands can be entered.

__ 11. Run some command line commands.

» Enter one or more of these: **xcalc &** ; **ls** ; **ps**

__ 12. Start the Desktop Terminal **dtterm**, using the Personal Applications Front Panel pop-up menu (the control that looks like a piece of paper and a pencil).

» Click the arrow above the Personal Applications Front Panel control.

The Personal Applications control is on the left side of the control panel, between the File Manager and Mail controls.

» Click the `Terminal` menu item. This will display a **dtterm.**

__ 13. Run some command line commands.

» Enter one or more of these: **xcalc &** ; **ls** ; **ps** .

__ 14. Compare the **aixterm** and the **dtterm** windows. What differences do you see?

» **dtterm** has a window menu, edit menu, options menu, help menu and scroll bar.

» There is hyperlinked Help for **dtterm.**

» **aixterm** will be on IBM AIX systems with X Windows installed, while **dtterm** should be on any UNIX platform with CDE installed.

__ 15. In the **dtterm** session, use the **Edit** menu bar option to copy and paste text.

» At the **dtterm** window shell prompt, enter: **ls -a**.

» Place your mouse pointer just before `.profile` and, while pressing the left mouse button, drag across the file name. Once the file name is reverse highlighted, release the mouse button. It should remain highlighted.

» Click on the **Edit** menu and then click **Copy** in the drop-down menu.

» At the command prompt, type in "**cat** " (followed by a space - do NOT press **Enter**).

» Click the **Edit** menu and then click **Paste** in the drop-down menu.

» You will see that the file name has been placed on your command line. Press **Enter**.

__ 16. Now close all open windows, except the Front Panel, and we will work with the File Manager.

Working with the File Manager

__ 17. Select the File Manager control from the Front Panel to access the File Manager.

» Click the icon that looks like the open drawer of a filing cabinet with a file folder tilted on its side, to the right of the calendar.

__ 18. Make sure that you are in your Home Directory: called /**home**/**teamxx**. The current directory is displayed at the top of the window.

» If you are not in that directory, you will need to navigate to that point:

» To navigate up the structure, either double-click (with the left mouse button) on the **go up ..** icon or double-click the directory name that you want to go to in the directory path shown in the window below the menu bar.

» When you navigate up and down the directory structure the content of the current directory is displayed.

» When you want to navigate down the structure, double-click the directory (folder) that you want to go to. Choose from the folders of those displayed in the window representing the contents of the current directory.

__ 19. View several of the files.

» Point at the icon representing the file, and then double-click with the left mouse button. Close any windows displaying the file contents.

__ 20. So that you have a few items to work with, the first thing you'll need to do is create a few new files.

» To create a new file entry in the current directory, click the **File** menu bar option. Then, click **New File**.

» Type in the name of the new file in the window presented and click **OK**. This creates a new empty file in the directory.

Do this several times to create several new files. You can use any names that you like as long as they don't conflict with anything that already exists. After you have done this you'll see several new entries in the current directory.

__ 21. Click on one of your new files and then click **Selected** in the menu bar. Click **Open** to edit the file. This will invoke the CDE Text Editor.

__ 22. Enter several lines of text in each file. Play with using the mouse pointer and/or cursor control keys to place your cursor in various locations in the text. Play with changing the text, using the insert and delete keyboard functions. You will notice that the CDE Text Editor is not the **vi** editor.

Content is immaterial, but for at least one of the files, create a small shell script.

When you have finished editing a file, save the file by clicking the **File** option in the menu bar, then selecting **Close**. Confirm that you want to save the file when that window is presented.

» If you cannot think of anything for a shell script then make it contain:
```
print Executing $0; date; print End of $0
```

» The Text Editor is much easier to use than **vi**. Use the Help System as needed.

__ 23. Add execute permission to the shell script you just created. Once this is complete, execute the shell script.

» Right click on the files icon.

» Choose the Change Permissions option from the pull down menu displayed. You can see what the current properties are - including File permissions.

» Click **Execute** permission for the owner. Click **OK** to commit the changes. The icon will change to a lightening bolt indicating the file can be executed.

__ 24. To execute the Shell Script, be sure its icon shows as a lightening bolt. Double-click the **Shell Script** icon. On the `Action:Run` window click **OK**. A window will appear showing the results of the Shell Script. Once you have reviewed the results, close the **Run** window.

Now you have a number of files that you can use in some drag and drop operations.

Drag and Drop Operations

You will need to be working with the files in your Home directory, so these should be displayed in the File Manager window.

If you are not at the correct directory, navigate up and down the structure until you get to where you want/ought/need to be.

__ 25. Use the mouse to move one of the files in your `$HOME` directory to the workspace backdrop. This will create a shortcut to access the file.

The file icon has been dropped onto the backdrop and will stay there for fast and convenient access. Now, if the file is executable, use the left mouse button and double-click the file icon to make it run.

» Select a file by clicking it once with the left mouse button.

» Press and hold the left mouse button with the pointer on the selected file.

» Move the mouse around; the file icon outline will follow.

» Choose a free spot on the backdrop where you want to put the file icon.

» When you are there, release the mouse button.

__ 26. With the pointer on the file icon on the backdrop, press the right button on the mouse.

What actions can you take on the file?

» A menu pops up.

» Depending whether the file is a directory, text file or a executable binary, you can edit, execute, rename, open another view, or remove it from the desktop.

You can drag a selected file from the Directory display presented by the File Manager or from the desktop and place it somewhere else. You cannot place the same file more than once on the desktop backdrop. You cannot drop a file on itself.

__ 27. While dragging a file, take it across the controls on the desktop.

What do you see?

» You will see that some of the icons will highlight indicating that they will accept the file, for example, the printer and the trash can.

» Some will not highlight. For example, the Style Manager or the Workspace switches.

__ 28. Drop the file on the Clock control. What happens?

» If you try to drop an item to a control that will not accept it, it flies back home.

File Manager - Finding, Copying and Deleting Files

The Desktop File Manager is one of the most useful and powerful tools in CDE. This section explores more of the File Manager capabilities.

__ 29. Set the File Manager preferences to display a Directory Tree diagram, starting at the root directory.

» On the File Manager window, double-click the root directory icon.

» Click **View**.

» Click **Set View Options.**

» In Headers, select **Iconic Path**, **Text Path**, and **Message Line**
In Show, select **By Tree** and **Folders Only**
In Representation select **By Small Icons**

» Click **Apply**.

__ 30. Navigate to the root directory in the File Manager window.

» Double click (left mouse button) on the root directory in the iconic path near the top of the window.

__ 31. Expand the /**usr**/**dt** directory.

» Click the **+** in front of the **/usr** icon (you may have to scroll).

» Click the **+** in front of the **dt** icon (again, you may have to scroll to see the file names listed).

» The /**usr**/**dt** directory contains CDE executables and default configurations.

__ 32. Set your viewing options to see a single folder at a time (rather than a tree structure) and using small icons. Also request display of the full path using icons near the top of the window.

» In the Set View options window:

Select only **Iconic Path** in Headers;
Select **By Single Folder** in Show;
Select **By Small Icons** in Representation.
Click **Apply.**

__ 33. Set your viewing options to display by properties (such as modify date, permissions, owner etc). This output will look similar to the output of the **ls -la** command.

» In the Set View Options window:

Unselect all choices in Headers
Select **By Tree** and **Folders Only** in Show;
Select **by Name, date, size...** in Representation.
Click **Apply**.

__ 34. Close the File Manager and any windows that it opened.

__ 35. Use the File Manager to execute the **date** command. This command is found in the /**bin** directory.

» Click the **File Manager** icon.

» Double-click the root directory icon.

» Double-click the bin directory icon.

» Scroll and find `date`.

» Double-click **`date`**.

» From the `Action:Run` panel, click **`OK`**.

» After **`date`** executes, close the `Run Panel`.

» Close the `File Manager`.

__ 36. Use the File Manager to create the directory **cdelab** in your `$HOME` directory.

» Click the **`File Manager`** icon.

» Verify that the icon path shows /**home**/**teamxx** (if not, click **`File`** and choose **`Go Home`**).

» Click **`File`**.

» Click **`New Folder...`** (a folder is like a directory to the CDE File Manager).

» Enter new folder name: **cdelab**

» Click **`OK`**.

» Close the **`File Manager`**.

__ 37. The File Manager can also be used to execute a find operation. Use the File Manager to find all pixmap files (files with an extension of **.pm**) in the CDE /**usr**/**dt** directory.

» Click the **`File Manager`** icon.

» Click **`File`**.

» Click **`Find`**.

» Enter File or Folder Name: **`*.pm`**

» Enter Search Folder: **`/usr/dt`**

» Click **`Start`**.

__ 38. Copy two or more of the pixmap files to the **cdelab** subdirectory.

» Select a **.pm** file by clicking it.

» Click **`Put in Workspace`**.

» Drag the pixmap icon from the workspace to the **`cdelab`** icon by holding down the **`Ctrl`** key and the left mouse button. (Dragging the icon without using the **`Ctrl`** key is like doing a move.)

» When the pixmap is over the **`cdelab`** icon, release the mouse button and **`Ctrl`** key to drop the pixmap.

» Repeat these steps to copy two more pixmaps.

__ 39. Rename one of the files to **myicon.pm.**

- » Double-click the **cdelab** icon.
- » Click the filename of one of the pixmap icons.
- » When the mouse moves, the name of the file displays in reverse video. It is possible to type over the name to enter a new name. Enter **myicon.pm**
- » Press **Enter**.

__ 40. Delete the **myicon.pm** file using the mouse and the Front Panel trash can.

- » Click the **myicon.pm** file icon to select it.
- » While pressing the left mouse button, drag it to the Trash Can on the Front Panel. Release the mouse button.
- » Click **OK** on the Trash Can Warning to delete the file.

__ 41. Delete a second pixmap file using the File Manager Menu Bar.

- » Click a pixmap file icon in your directory.
- » Click **Selected**.
- » Click **Put In Trash**.
- » Click **OK** on the Trash Can Warning.

__ 42. With CDE it is possible to retrieve a deleted file. Restore **myicon.pm.**

- » Note that the Trash Can icon looks like it contains something. Click the **Trash Can** icon. This displays the Trash Can window.
- » Click the **myicon.pm** icon.
- » Click **File**.
- » Choose **Put Back** to retrieve a deleted file.

__ 43. Empty the trash can.

- » Select **File** on the `Trash Can` window.
- » Select **Select All**.
- » Select **File**.
- » Select **Shred**.
- » Click **OK** when the shred warning is displayed. The files will be deleted.
- » Close the `Trash Can` window.

__ 44. Change the Owner and Group permissions of the restored file to read/write.

- » Click the **myicon.pm** icon in the File Manager window.
- » Click **Selected**.
- » Click **Change Permissions**.

» Click **Write** for Owner.

» Click **Write** for Group.

» Click **OK**.

__ 45. Close the File Manager.

__ 46. At this point, you may continue with the optional exercise or exit out of CDE. If you are in a VNC environment, do not use the Exit icon, but instead just lock the session. If completely done with using the CDE interface go to the last step in this exercise and close down the interface.
Skip this step to perform the optional steps.

» If in a VNC environment, click on the Lock icon.

» Click **Exit** on the Front Panel.

» Click **Continue Logout** on the confirmation panel.

Optional Exercise Steps

__ 47. Click the Calendar control on the Front Panel. Add an appointment in the next week.

» Click the **Calendar** control on the Front Panel; it is just to the right of the clock.

» Click the **Appointment Editor** on the Calendar Toolbar. The Appointment Editor is the first icon on the left.

» In the Date window type a date for next week.

» In the Start time window, click the rectangle button to view the various start times. Click a start time. Click **AM** or **PM** as necessary.

» In the End time window, click the rectangle button to view the various end times. Click an end time. Click **AM** or **PM** as necessary.

» In the What window type what the appointment is for.

» Click **Insert**.

» Click **Cancel** to close the Appointment Editor window.

» Your appointment should now be displayed on the month-view calendar. If the appointment is made for the next month, click the > to the right of Today on the Calendar Toolbar to view the next month.

__ 48. Change the view to Day View to view the appointment you have scheduled.

» Use the left mouse button to click the day (on the month view calendar) that the appointment has been scheduled.

» Click the **Day View** icon on the Calendar Toolbar. The Day View icon is the fourth icon from the right.

__ 49. Change the view to Week View to view the appointment you have scheduled.

» Click the **Week View** icon on the Calendar Toolbar. The Week View icon is the third icon from the right.

__ 50. Set a reminder to yourself for the appointment. Make the appointment private so that others cannot view it on your calendar.

» Click the **Appointment Editor** icon. Again, this icon is the first icon on the Calendar Toolbar.

» Be sure your appointment shows in the Time What window. If it does not, cancel the Appointment Editor and then click the day your appointment is scheduled. Once this day is highlighted, click the **Appointment Editor**.

» Select your appointment in the Time What window

» On the Appointment Editor, click **More**. An extended appointment window will be displayed.

» Under Reminders, choose how you would like to receive the reminder; by beep, flash, popup or mail.

» Click **Privacy** and choose a preferred privacy option.

» Once the reminder is complete, click **Change**. Then click **Cancel**.

__ 51. Return to the *month view* icon on the calendar menu bar.

» Click the **Month View** icon on the calendar menu bar. It is the second from the right.

__ 52. Close the Calendar window

» Click the upper left of the window to display the menu window. Click **Close**.

__ 53. Exit out of the vnc session.

» If using a VNC session, close the desktop window. Then switch to your ASCII terminal session and type **vncserver -kill :X** where **X** is the session number you previously created.

END OF EXERCISE

Appendix A. Customizing AIXwindows (1)

(with Hints)

Estimated Time

00:45

What This Exercise Is About

This exercise shows the students how they can customize their AIXwindows environment.

What You Should Be Able to Do

At the end of the lab, you should be able to:

- Customize the **.xinitrc** file
- Customize the **.Xdefaults** file

Introduction

In this exercise, students will learn how to edit files to customize their AIXwindows environment.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- If VNC is used to provide the AIXWindows environment, then the file that controls the AIXWindows session initialization is called **xstartup**, and is located in **.vnc** subdirectory of the user's home directory.
- If the lab exercises is being done on a local attached graphics console (LFT), then follow the instructions marked **(LFT)**. If a VNC server session is being used to display AIXWindows, then follow the instructions marked **(VNC)**.
- All hints are marked by a » sign.

Customizing the .xinitrc File

The **.xinitrc** file is used by the **startx** shell script to initialize the AIXwindows session. (Actually, **startx** executes **xinit**, which reads the **.xinitrc** file.) If VNC is used to provide the AIXWindows environment, then the file that controls the AIXWindows session initialization is called **xstartup**, and is located in **.vnc** subdirectory of the user's home directory.

__ 1. Log in to your AIX system, either locally or through a terminal emulator.

» login: **teamxx**

» passwd: <your password>

__ 2. If using a local graphics display (LFT), copy the file **/usr/lpp/X11/defaults/xinitrc** into your $HOME directory and call the file **.xinitrc.** If using a VNC session, there is no need to copy the file; the **xstartup** file is already present in **$HOME/.vnc**.

»**(LFT) cp /usr/lpp/X11/defaults/xinitrc ~/.xinitrc**

__ 3. Edit the file and make the following changes:

- Add a second hand to the **xclock**.
- Make the root window solid black.
- Add, on a new line before the **exec mwm** line, the following:
  aixterm -T "Bills Window" &

»**(LFT) vi .xinitrc**

» **(VNC) vi xstartup**

» Update the **xclock** command to include the option **-update 1**

» Update the **xsetroot** command to look like this: **xsetroot -solid black**

» Add the following: **aixterm -T "Bills Window" &**

» Save the file using **<Esc> :wq**

__ 4. Start the AIXwindows session. If you are accessing AIXWindows through a VNC client, then start the VNC server with **vncserver**, and connect to the specified session with the client. Does the AIXwindows environment look different? It should!

» **(LFT) startx**

» **(VNC) vncserver**

Customizing the .Xdefaults File

__ 5. Execute the command **aixterm -keywords | pg** to view all the resources that can be customized for an **aixterm** window.

» **aixterm -keywords | pg**

__ 6. Create the **.Xdefaults** file in your $HOME directory and add the following resource definitions:

```
Aixterm*foreground:    DarkSlateGrey
Aixterm*background:    wheat
Aixterm*geometry:      80x30
Aixterm*font:          rom10.iso1
```

» **vi .Xdefaults**

» Add the above lines into the file. Be sure there are no trailing blanks after any of the entries. Save the file using **<Esc> :wq**

__ 7. Restart AIXwindows. This will cause your new **.Xdefaults** file to be read and used for any new **aixterm** windows you create. Now, open a new **aixterm** window. Does it have the characteristics specified in the **.Xdefaults** file?

» Move the mouse to the root window and press the right mouse button. This will display the root menu.

» Keeping the right mouse button depressed, move the mouse pointer to Restart... and release the mouse button.

» When asked if you want to Restart Mwm, use the left mouse button to click **OK**.

» Using the left mouse button, click one of your **aixterm** windows so that it becomes the active window.

» On the command line enter: **aixterm &**. This new window should use the characteristics you entered into the **.Xdefaults** file.

__ 8. Now, end the AIXwindows session and then restart it. If using a VNC environment, switch to your ASCII terminal session and issue **vncserver -kill :*session*** where

session is the VNC session ID. Restart the session by running **vncserver**. What do the two original windows look like? Why?

» **(LFT) <Ctrl> <Alt> <Backspace>** to end AIXwindows

» **(LFT)** $ **startx**

» **(VNC) vncserver -kill :X (where X is the session ID)**

» **(VNC) vncserver**

__ 9. Edit the **.Xdefaults** file and update the following lines for new colors:

```
Aixterm*foreground:   grey
Aixterm*background:   navy
```

» **vi .Xdefaults**

» Edit the file to change the colors for your **aixterm** windows. Save the changes.

__ 10. Restart the mwm and then create a new **aixterm** window from the command line. Does it use your new color specifications? It should!

» Move the mouse to the root window and press the right mouse button. This will display the root menu.

» Keeping the right mouse button depressed, move the mouse pointer to **Restart...** and release the mouse button.

» When asked if you want to Restart Mwm, use the left mouse button to click **OK**.

» Using the left mouse button, click one of your **aixterm** windows so that it becomes the active window.

» On the command line enter: **aixterm &**

__ 11. Exit your AIXwindows environment and log out from your system. If using VNC, switch to your ASCII terminal session and kill the VNC server.

» **(LFT)** Press **<CTRL><ALT><BACKSPACE>**.

» **(VNC) $ vncserver -kill :X** (where X is the VNC session ID)

END OF EXERCISE

Appendix B. Customizing AIXwindows (2)

(with Hints)

Estimated Time

00:40

What This Exercise Is About

This exercise shows the students how they can customize their AIXwindows environment.

What You Should Be Able to Do

At the end of the lab, you should be able to:

- Use the **custom** tool to tailor colors and fonts
- Use the **custom** tool to tailor size, location, icons, and the scrollbar
- Customize the Motif window manager (mwm)
- Use the **xsetroot** command to customize the root window

Introduction

In this exercise, students will learn how to use the AIXwindows **custom** tool to customize their AIXwindows environment.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- If the lab exercises is being done on a local attached graphics console (LFT), then follow the instructions marked **(LFT)**. If a VNC server session is being used to display AIXWindows, then follow the instructions marked **(VNC)**.
- All hints are marked by a » sign.

Using the Custom Tool: Color and Fonts

__ 1. Log in to your system and start AIXwindows. If the AIXWindows session is being accessed through VNC, start the VNC server, and access it from your VNC client.

» **(LFT) $ startx**

» **(VNC) $ vncserver**

__ 2. Make sure you have two **aixterm** windows open as well as the **xclock**. Also, start the scientific calculator.

» **xcalc &**

__ 3. Start the AIXwindows customization tool.

» Move the mouse pointer to the root window and press the right mouse button. While still holding down the right mouse button, point to the **Custom** option and release the mouse button. The **Customizing Tool** will appear.

__ 4. On the Customizing Tool window, choose **xcalc**.

» With the left mouse button, point and click `xcalc`. The line should be highlighted.

» Use the left mouse button to click `OK`.

__ 5. View the different resource categories that can be changed for the **xcalc** application. What sorts of resources can be changed? Choose **Colors**, which is the default resource category.

» Use the left mouse button to click the small rectangle in the **Colors** box. This will display the other resource categories that can be customized for this application.

» Use the left mouse button to click **Colors.**

__ 6. Change the background color for **xcalc** to the color of your choice.

» Move the mouse to the line titled `window interior background` and use the left mouse button to click the **Colors...** box. The Colors browser will be displayed.

» There are two ways to select a color. One way is to use the left mouse button to scroll through the various colors. When you find one that looks interesting, click the color with the left mouse button. The color will be displayed. Note the red, green, and blue sliders in the window will change based on the color chosen.

» Another way to choose a color is to use the left mouse button to slide the red, green and blue bars to whatever color mixture you want. The color will be displayed. Once you decide on a color, click Match RGB to Closest Color Name, and review the results.

» Click **Apply**. The background color of the **xcalc** should change.

» Click **OK** to close the window.

__ 7. Switch focus to an **aixterm** window and display the contents of **.Xdefaults**. Has it been updated? It should not have been!

» Use the left mouse button to click the **aixterm** window.

» **cat .Xdefaults**

__ 8. So, to have your values saved in **.Xdefaults**, change your focus back to the **xcalc** customizing window. Save the values you have chosen.

» Use the left mouse button to click back to the xcalc Customizing Window.

» Click **File**, which is located in the upper left of the window.

» Click **Save As...**.

» On the Save As... window, you are given the opportunity to choose which file you wish to save the values in. The default is **$HOME/.Xdefaults**. Click **OK**.

__ 9. Now, review the **.Xdefaults** file again. Your resource change should now be there.

» Use the left mouse button to click the **aixterm** window.

» **cat .Xdefaults**

__ 10. Return to the **xcalc** Customizing window and now choose the resource category of Fonts.

» Use the left mouse button to click the Customizing Tool window.

» Click the small rectangle in the Colors box. Keep the left mouse button pressed.

» Point to **Fonts** and release the mouse button.

__ 11. View the various fonts that can be used for the window interior.

» Click the **Fonts...** button which corresponds to the window interior option.

» The Fonts browser will appear.

__ 12. The List of Fonts window is used to display all the possible fonts. Feel free to scroll through them, but be aware that there are LOTS of fonts in the list! You can narrow down the list of fonts by choosing **Family**, **Weight**, **Slant**, **Style**, **Spacing**, and **Size** in the respective selection windows. Below these windows will be feedback indicating how many fonts match the selection criteria.

Click a font from the **List of Fonts** that appears interesting. It will be displayed in the Sample box (some fonts will not display). If you have trouble finding a font you like, try the following to narrow down the search:

```
Family: Helvetica
Weight: Bold
Slant: All
Style: All
Spacing: All
Size: 14
```

Choose a font to be used for the **xcalc** window and save your choice as you did for the background color. Verify they change has been added to your **.Xdefaults** file. Close the Customizing windows.

» Use the left mouse button to make your font choice. Once you select a font, it will be displayed in the Sample box.

» Once you have decided on the font to use, be sure it is highlighted and then click **Apply**. **xcalc** should now use the new font.

» Click **OK**.

» On the **xcalc** customizing window, click **File**.

» Click **Save As...**

» Click **OK** to save your changes to the **.Xdefaults** file.

» Click the upper left of the **xcalc** Customizing Tool window to open the window menu and then click **Close**.

» Run **cat .Xdefaults** to verify the font information has been updated in the file.

__ 13. Use the customizing tool to change the background color for an **aixterm**. When you choose Apply will the color of your existing **aixterm** windows change like it did for the **xcalc** window? Will the new color be updated in the **.Xdefaults** file? Verify that your change updated **.Xdefaults** and affects the appearance of a new **aixterm**.

» Move the mouse point to the root window and hold down the right mouse button. While still holding down the button, point to Custom and release the button.

» On the Customizing Tool window click **aixterm** and then **OK**.

» Click the **Colors...** box for window interior background.

» On the Colors window, choose any color and then click **OK**.

» On the aixterm Customizing window, click **File**. Then, click **Save as...**. On the Save As... window, click **OK**. Your changes have now been added to the **.Xdefaults** file.

» Close the **aixterm** Customizing window by clicking in the upper left corner to open the window menu and then click **Close**.

» **cat .Xdefaults**

» **aixterm &**

Customizing the root Window with the **xsetroot** *Command*

We will next change the root menu. This is done using the **xsetroot** command from the command line of one of your **aixterm** windows.

__ 14. Change the root window to solid blue.

» **xsetroot -solid blue**

__ 15. Change the cursor pointer to a skull and crossbones (called pirate), to a shuttle, or to gumby. Move the cursor to the root window to view the new cursor shape.

» **xsetroot -cursor_name pirate**

» Move the cursor to the root window to view the new cursor shape.

» **xsetroot -cursor_name shuttle**

» **xsetroot -cursor_name gumby**

__ 16. Have the root window display xsnow (snowflakes) or escherknots - take your pick. These bitmap images are found in the directory **/usr/include/X11/bitmaps**. You may wish to view the file names in this directory for other bitmaps of interest. The bitmaps themselves are black and white images, so you may want to set other colors for the background and foreground.

» **xsetroot -bg white -fg pink -bitmap /usr/include/X11/bitmaps/xsnow**

» **xsetroot -bg lightblue -fg navy -bitmap \ /usr/include/X11/bitmaps/escherknot**

__ 17. If you decide you like any of these root window options, how would you make your customization permanent, that is, available every time you start AIXwindows?

» Change the **xsetroot** command in **.xinitrc**

Optional Exercises

Using the Custom Tool: Size and Location, Icons and Scrollbar

__ 18. Make sure you have a running Calculator Tool. If not, start one.

» In your **aixterm** window enter: **xcalc &**

__ 19. Start the AIXwindows Custom Tool and choose **xcalc** again.

» Move the mouse pointer to the root window and press the right mouse button. While still holding the button down, point to Custom and release the mouse button.

» On the Customizing Tool window, click **xcalc** and then **OK**.

__ 20. Choose the Size and Location resource category and customize the size of the **xcalc**.

» On the xcalc Customizing window, use the left mouse button to click the small rectangle in the **Colors** box. Click **Size and Location**.

» On the Size and Location window, try using different pixel values for height and width. As a suggestion, start with a size of **300x400**. Press **Enter** after choosing the sizes you want. Notice the change in the calculator tool.

» Save the size values if you want by clicking **File**. Then, click **Save as...**. On the Save As... window, click **OK**. Your changes have now been added to the **.Xdefaults** file.

__ 21. Suppose you wish to update the icon used for a particular AIXwindows application. To demonstrate how this is done, we will change the icon used for **xcalc**. You may first want to iconify and then restore the **xcalc** window to view the icon that is used. Then, use the **xcalc** Customizing window, and choose the icon resource category.

» On the xcalc Customizing window, click the small rectangle in the **Colors** box and then click **Icon**.

__ 22. Choose a new icon for the **xcalc** window: have the icon look like a terminal. Once you have completed this task, review the **.Xdefaults** file to verify that your entry has been added. Test the new icon to verify that it is being used.

» Click **Pictures** which corresponds to the icon picture * line.

» The window under Files lists all the available pictures that can be used as an icon. Scroll through the list to see what the options are. To view any of them, click the file name and then on **View Picture**. The escherknot is an interesting icon to view.

» Under **Files**, click the icon file named “terminal” and then **View Picture**. Once you approve of this choice, choose Cancel to remove the picture.

» Click **OK** to save your choice.

- » On the **xcalc** Customizing window, use the left mouse button to click **File**, then **Save As...**, then **OK** to save your values in the **.Xdefaults** file.
- » Close the Customizing Tool window by choosing **Close** from the window menu.
- » Now, from an **aixterm** window, view the **.Xdefaults** file using the **cat .Xdefaults** command to make sure your change has been added.
- » In order for the new icon to be used, MWM must be restarted. Move the mouse pointer to the root window and press the right mouse button. Choose **Restart** and then **OK** to restart the Motif window manager.
- » Now, iconify the **xcalc** window. It should use the new icon you have chosen.

__ 23. Now, add a scroll bar to the **aixterm** windows. Verify that the **.Xdefaults** file has been updated and test to verify that the scroll bar works.

- » Move the mouse pointer to the root window and press the right mouse button. Holding down on the button, point to **Custom** and release the button.
- » On the Customizing Tool window, click **aixterm** and then **OK**.
- » Click the small rectangle in the Colors box and then click **Scroll Bar**.
- » Click the box for **visible scroll bar** and choose **true**.
- » Click **File**, **Save As...** and then **OK**.
- » Close the Customizing Tool window by choosing **Close** from the window menu.
- » View the **.Xdefaults** file using the **cat .Xdefaults** command. The scrollbar resource should be listed.
- » Start an **aixterm** window using the **aixterm &** command. The new window should display a scrollbar.

__ 24. In your new **aixterm** window, list the files in /**usr**/**bin** and then use the scrollbar to go back and forth in the listing.

- » In the **aixterm** window run: **ls /usr/bin**
- » In the scrollbar area click your right mouse button to scroll up
- » In the scrollbar area click your left mouse button to scroll down

Customizing the Motif Window Manager (MWM)

__ 25. Use the AIXwindows **custom** tool to update the MWM with the following characteristics:

```
window manager background: red
window manager foreground: blue
```

Verify that **.Xdefaults** has been updated.

- » Move the mouse pointer to the root window and press the right mouse button. On the root menu, click **Custom**.
- » On the Customizing Tool window click **mwm** and then on **OK**.
- » From the Mwm Customizing window, view the various resource categories by moving the left mouse button to the small rectangle in the **Colors** box and clicking. You will see that there are many resources that can be tailored. Choose **Colors**, which is the default.
- » For the window manager background type in **red**.
- » For the window manager foreground type in **blue**.
- » Click **File**, **Save As...** and then **OK** to save the new resource values in the **.Xdefaults** file.
- » Change the focus to an **aixterm** and view the **.Xdefaults** file using the **cat .Xdefaults** command to verify the changes have been stored.
- » Move the mouse pointer to the root window and click the right mouse button. Choose **Restart** from the root menu and then **OK** to restart the mwm. What happens?
- » Now, view both the root menu and the window menu. The colors should have changed!

__ 26. Some users prefer to use the pointer focus policy so they don't have to click a window to make it the active window. The pointer focus policy allows you to merely move the pointer to a window to make it the active window. If you are interested, change your focus policy to pointer. Verify that **.Xdefaults** has been updated and that the new focus policy works.

- » From the Mwm Customizing window, click the small rectangle in the **Colors** box. Then, click **Focus.**
- » Move the pointer to the box that corresponds to the keyboard focus policy and click the small rectangle.
- » The default is explicit, meaning that you need to click a window to make it the active window. If you wish to change the focus policy, click pointer.
- » To save this change, click **File**, **Save As...** and **OK**.
- » Use the **cat .Xdefaults** command the verify the changes have been made to the **.Xdefaults** file.
- » Now, restart the mwm by moving the mouse pointer to the root window and using the right mouse button, point to **Restart...**. Click **OK** to restart the Motif Window Manager.
- » You will notice now that when you move the mouse around, the different windows will be highlighted. The highlighted window is the active window.

The pointer focus policy seems to work best if the windows are not overlapped.

» End the customizing tool when you have finished. You may also want to iconify or close some windows if your screen is looking cluttered.

END OF EXERCISE

Appendix C. Customizing CDE

(with Hints)

Estimated Time

01:00

What This Exercise Is About

This exercise provides an opportunity to customize the CDE Desktop.

What You Should Be Able to Do

At the end of the lab, students should be able to:

- Customize CDE using the Style Manager
- Customize the Front Panel

Introduction

Students will work as teams using a graphics terminal to customize their CDE environment. This machine exercise will focus on using the interactive customization features of CDE. First, the CDE environment will be customized using the Style Manager. Then, the Front Panel will be customized.

Exercise Instructions with Hints

Preface

- All exercises for this unit depend on the availability of specific equipment in your classroom.
- All hints are marked by a » sign.

Customizing the Front Panel

__ 1. If logging in through a VNC client, first connect to the AIX machine via an ASCII terminal session and log in as your user. Start the VNC server session by typing **vncserver**. Switch back to your desktop, and start the VNC client application, specifying the hostname and VNC session number.

If logging in on a graphics console (LFT), log in as your userid.

» Connect and log in as your userid.

»`$ vncserver`

» Switch back to your desktop, and launch the VNC client application. Enter the machine's name and VNC session number and password when required. The VNC client window should appear with the CDE environment running within.

__ 2. Customize your Workspaces as follows:

Rename each Workspace.

» Single-click a Workspace Switch button to select that Workspace.

» Double-click this same Workspace Switch Button to change its name.

» Type in a new name and press **Enter** to complete the change.

Change the Backdrop of each Workspace.

» Click the **Style Manager** icon.

» Click the **Backdrop** icon.

» Select your choice of backdrop and click **Apply**.

» Click **Close**.

» Select another workspace and repeat.

Turn on the screen saver and screen lock.

» While in the Style Manager, click the **Screen** icon.

» Make your choices and click **OK**.

Set the window behavior.

» While in the Style Manager, click the **Window** icon.

» Make your choices and click **OK**.

Select a different palette for the workspaces.

» In the Style Manager, click the **Color** icon.

» Select a palette and click **OK**.

__ 3. Add a fifth workspace and customize its style using the Style Manager.

» Click a workspace button with the right mouse button and select **Add Workspace**.

» Use the Style Manager to make any changes that you want.

__ 4. Set the new session as your Home session, and set Startup to return to your Home session at login.

Note: This is not supported in a VNC session. If using VNC, skip to step 6.

» Click the **New Workspace** Button.

» In the Style Manager, select the **StartUp** icon.

» Select **Set Home Session...**

» Click **OK** to replace.

» Select **At login, Return to Home Session**.

» Click **OK**.

__ 5. Log out and log in again. Check to see that the state of your session matches what you set in the previous steps.

» Click **Exit**.

» Click **Continue logout** when asked to confirm.

» At the Login Manager panel, log back in.

__ 6. Add the same **dtterm** session to all workspaces.

An application can be assigned to one or more workspaces by using the **Window** button menu.

» If you do not have a **dtterm** session started, raise the Personal Applications subpanel and click **Terminal**.

» Click the **dtterm's** window menu button at the top left of the window frame (the dash).

» Note the options on the window menu. Click **Occupy All Workspaces** to place the application in all workspaces. (If you had wished to remove an application from a workspace, you could have chosen **Unoccupy Workspace**.)

__ 7. Use the **ls** command in the **dtterm** to list the current directory.

Check each of the workspaces to see if the same application session is available.

__ 8. Now, remove a Workspace application from one or more Workspaces.

» If you want to remove an application, which you previously occupied into multiple Workspaces, from a Workspace you can see that there is an `Unoccupy Workspace` menu item in the pull down Window menu for just that purpose.

__ 9. Have the **dtterm** application appear on the Front Panel as the default application associated with the `Personal Applications` control.

» Click the arrow above the **Personal Applications** control to display the subpanel.

» Point at **Terminal**, the item you want to have on the Front Panel.

» Press the right mouse button.

» Choose **Copy to Main Panel**.

» The icon for the terminal should now appear on the Front Panel.

» Click the arrow above the **Personal Applications** control to close the subpanel.

__ 10. Tear off the Personal Applications subpanel menu, and place it on the workspace.

» Click the arrow above the **Personal Applications** control to raise its subpanel.

» With the subpanel now raised, point at its title bar.

» Press and hold the left button on the mouse and drag the whole menu to a convenient location on the backdrop.

» Release the mouse buttons to drop the menu at that location.

» The menu will stay displayed after an item has been selected. Normally, it will close after one of its items has been selected.

__ 11. Create a new subpanel for the `Style Manager` control and add the `Icon Editor` and the **aixterm** applications to it.

» Point at the `Style Manager` control. Notice that it does not have a subpanel since there is no arrow above its control.

» Press the right mouse button to get a pop-up menu.

» Select the **Add Subpanel** option.

» If the control already had a subpanel present, there would have also been an option to delete the subpanel.

» Click the subpanel. Note that it contains two items: **Install Icon**, which enables you to add more items to this subpanel, and a function related to the control itself.

» Click the **Application Manager** control on the Front Panel.

» Select **Desktop_Apps**.

» Click with the right mouse button to display a pull-down menu.

» Select **Open In Place**.

» Point at the scrollbar at the right, press and hold the left mouse button, pull down until the Icon Editor entry shows in the window. Release the mouse button.

» Click the **Icon Editor**.

» Point at the Icon Editor again and drag the outline to the popped up subpanel. Drop the icon onto the Install Icon control.

» The Icon Editor is added to the subpanel.

» Close the Application Manager window.

» The **aixterm** icon is in the **Desktop_Tools** directory of the Application Manager. Click the **Application Manager** control.

» Click **Desktop_Tools**.

» While pointing at Desktop_Tools, press the right mouse button to display a pull-down menu.

» On the menu, choose **Open in Place**.

» Use the scrollbar to locate **aixterm** and then use the left mouse button to drag it to the Install Icon control on the subpanel.

» Now, there should be two new items on the Style Manager's subpanel.

__ 12. Now, remove the Icon Editor from the new subpanel.

» On the Style Manager's subpanel, point to the item you wish to remove (the Icon Editor).

» Press the right mouse button.

» Select **Delete**.

» Select **OK** to confirm.

Adding a New Control to the Front Panel

__ 13. Start the Application Manager.

» Click the **Application Manager** icon.

__ 14. Open the Personal Applications subpanel.

» Click the arrow in the Front Panel to open the subpanel.

__ 15. Drag the icon for **Firefox** from the Application Manager window on to the **Install Icon** from the Personal Applications subpanel.

» Select the **Firefox** icon, and drag it with the left mouse button on to the Install Icon.

__ 16. Close the `Personal Applications` subpanel.

» Click the arrow in the subpanel.

__ 17. Find out the name of the definition file in directory **$HOME/.dt/types/fp_dynamic**. Write down the file name:

» $ **ls $HOME/.dt/types/fp_dynamic**

» The file name should be **Firefox1.fp**

__ 18. Copy this definition file to directory **$HOME/.dt/types** and specify a new file name.

» $ **cd $HOME/.dt/types**

» $ **cp fp_dynamic/firefox1.fp browser.fp**

__ 19. Anchor the application control in the Front Panel by editing the copied definition file. Use your student notes to find out which lines must be changed.

» $ **vi browser.fp**

```
CONTAINER_TYPE    BOX
CONTAINER_NAME    Top
POSITION_HINTS    last
```

__ 20. Restart the CDE. After restarting CDE, you should see the application icon on the Front Panel.

END OF EXERCISE