AIX 5L Basics

(Course Code AU13)

Student Exercises

ERC 7.0

IBM IT Training Services
Worldwide Certified 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®
Hummingbird®	IBM®	Language Environment®
MVS™	OS/2®	PAL®
PS/2®	pSeries™	RISC System/6000®
RS/6000®		

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

Windows is a trademark 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 and other countries.

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

January 2003 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®
Hummingbird®	IBM®	Language Environment®
MVS™	OS/2®	PAL®
PS/2®	pSeries™	RISC System/6000®
RS/6000®		

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

Windows is a trademark 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 and other countries.

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

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/or 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 don't 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/or 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**.

Exercise 1. Using the System

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.

Exercise Instructions

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** etc.

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.

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

Basic Commands

__ 3. Display the system's date.

__ 4. Display the whole calendar for the year 2003.

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

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

__ 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.

__ 8. Display just your login name.

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

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

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

Send And Receive Mail

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

__ 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.

__ 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.

Communicating with other users

__ 15. Send a note to all users on the system indicating that you 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.

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.

__ 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 the five characters 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.

__ 19. Log off the system.

END OF EXERCISE

Exercise Instructions With Hints

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** etc.

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 2003.

- **$ cal 2003**

__ 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.

- **$ whoami**

__ 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 the five characters 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

Solutions

Following are the solutions for those instructions that include questions:

7. 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!

8. 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 2. AIX Documentation

What This Exercise Is About

The purpose of this exercise is to give the student the opportunity to explore and experiment with the **man** command and with the AIX V5.2 online documentation.

What You Should Be Able to Do

After completing this exercise, students should be able to:

- Execute the **man** command
- Initiate Netscape to access AIX online documentation
- Access documentation via the Tasks/Topics view, the Books view and using the Commands Reference
- Use the online Glossary
- Use the AIX Documentation Library

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 the Netscape Navigator to access AIX V5.2 online documentation. In order to perform these steps, it will be necessary to use graphics mode. If you are sharing a graphics terminal with another student, be sure to take turns performing the various steps in the exercise.

In order to access the Netscape Navigator, you will first need to access AIXwindows. This is done using the **xinit** command. More on AIXwindows will be covered later in the course.

In order to bring up the online documentation from the Netscape Navigator, you will need to know the name of your system. Use the **hostname** command to display this information. Your system name will be something like **sysx**, where **x** will be a unique number to identify your system.

Exercise Instructions

Introduction to online documentation: The internal organization of the AIX online documentation differs between AIX V5.1 and AIX V5.2. If you have access to the internet you will find the online documentation on this URL:

http://techsupport.services.ibm.com/server/library

man Pages

__ 1. 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>**.

__ 2. Using the **man** command, search on the keyword **calendar**. From the list produced, find the command that displays a calendar.

__ 3. Having found the **cal** command from the previous step, use **man** without any options to obtain the correct syntax of the command.

AIX online documentation

__ 4. Use the **hostname** command to find out the name of your system. It is necessary to know your system name in order to access the AIX online documentation. On AIXserver execute *xinit* or *startx* to start AIXwindows. Start the Netscape Navigator. Once the Netscape Navigator has been started, type in the following URL to access the AIX Documentation Library:
http://*HOST*/cgi-bin/ds_form (where ***HOST*** is the hostname of your system)
If Netscape presents any error windows, just click on **OK** and continue with the exercise. **Note**: An alternate method of accessing the Documentation Library is by using the **docsearch** command from the command line.

__ 5. So that you don't have to type this lengthy URL every time you wish to access the online documentation, add the URL to your list of bookmarks. Now, the next time you need to access your online documentation, just click on **Bookmarks** and then click on the bookmark labeled **Documentation Library**.

__ 6. Enlarge the font used by the online documentation to something that you prefer.

__ 7. Use the **Tasks and Topics** view to find information *User how-to's* i.e. how to change your password. When using the online documentation, click on **Back** (in the upper left of the window) to go back just one screen. Click on Library **Home** to return to the AIX V5.2 Documentation Library.

__ 8. Use the **Books** view to access the *AIX V5.2 System User's Guide: Operating System and Devices* manual.

__ 9. Use the **Commands** view to view information about a command.

__ 10. Use the **Glossary** to view the output of any term you would like clarification on. As an example, find out the meaning of the word *hypertext*.

__ 11. Use the advanced search function to find information. For example, let's say you wish to find information about the **lpstat** command and the word **printing**.

__ 12. From the AIX V5.2 Documentation Library, review what you have done during this session. You will have the ability to immediately re-access any of the documentation windows you have used so far in this session.

__ 13. Exit from the AIX V5.2 Documentation. If you were to continue running in the AIXwindows environment, you could also choose to iconify your documentation window.

END OF EXERCISE

Exercise Instructions - With Hints

Introduction to online documentation: The internal organization of the AIX online documentation differs between AIX V5.1 and AIX V5.2. If you have access to the internet you will find the online documentation on this URL:

http://techsupport.services.ibm.com/server/library

man Pages

__ 1. 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**

__ 2. Using the **man** command, search on the keyword **calendar**. From the list produced, find the command that displays a calendar.

- **$ man -k calendar**

__ 3. 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 online documentation

__ 4. Use the **hostname** command to find out the name of your system. It is necessary to know your system name in order to access the AIX online documentation. On AIXserver execute *xinit* or *startx* to start AIXwindows. Start the Netscape Navigator. Once the Netscape Navigator has been started, type in the following URL to access the AIX Documentation Library:
http://*HOST*/cgi-bin/ds_form (where ***HOST*** is the hostname of your system)
If Netscape presents any error windows, just click on **OK** and continue with the exercise. **Note**: An alternate method of accessing the Documentation Library is by using the **docsearch** command from the command line.

- $ **hostname** Record the results
- $ **xinit** This will bring up AIXwindows
- $ **netscape** This will bring up the Netscape Navigator
- On the Netscape URL type:
  - **http://*HOST*/cgi-bin/ds_form** (where ***HOST*** is the hostname of your system)

- Press **enter**

__ 5. So that you don't have to type this lengthy URL every time you wish to access the online documentation, add the URL to your list of bookmarks.

- Click on **Bookmarks**
- Click on **Add Bookmark**

  Now, the next time you need to access your online documentation, just click on **Bookmarks** and then click on the bookmark labeled **Documentation Library**.

__ 6. Enlarge the font used by the online documentation to something that you prefer.

- Click on **Edit**, which is found in the upper portion of the window, on the Netscape menu bar
- Click on **Preferences**
- Click on **Fonts** in the Category box
- To the right of the 'Variable Width Fonts' box, click on the small square in the box next to **Size**
- Click on a larger size font. **18.0** is a good size. When you click on a size, you will see the font change immediately
- When you find a font size you like, click on **OK**

__ 7. Use the **Tasks and Topics** view to find information *User how-to's* i.e. how to change your password. When using the online documentation, click on **Back** (in the upper left of the window) to go back just one screen. Click on Library **Home** to return to the AIX V5.2 Documentation Library.

- Click on the **Tasks and Topics** view tab
- Expand **User how-to's**
- Click on **How do I use some common commands and utilities?**
- Click on **Changing Passwords (passwd Command)**
- Use the scroll bar to the right of the window to read through the information. When you have read what you like, click on **Back two times**.
- You are now on the 'Task and Topics' screen. If time permits, feel free to explore any other topics listed here. When you are finished, click on **Library Home** (at the top of the documentation window) to return to the documentation home page.

__ 8. Use the **Books** view to access the *AIX V5.2 System User's Guide: Operating System and Devices* manual.

- Click on the **Books** view tab
- Expand **AIX 5L Version 5.2 Books**

- Expand **System User's Guide**
- Expand **System User's Guide: Operating System and Devices**
- Click on **Printers, Print Jobs and Queues**. This will show the various sections in the chapter.
- Click on a section of interest. You will notice that the text is displayed in the larger reading window. Scroll through the output to review the information available. When you have completed your review, click on **Back**.
- Click on **Library Home** to return to the Documentation Library. If time permits, feel free to browse additional online manuals.

__ 9. Use the **Commands** view to view information about a command.

- Click on the **Commands** view tab
- Expand **Alphabetical List of Commands**
- Expand the first letter of the command you wish to review. For example, if you wish to view information about the **wc** command, click on **W**.
- Scroll through the command list until you find the command you are interested in. Point and click on that command.
- Scroll through the output of the command. If time permits, click on **Back** and view the output of other commands.
- When you have finished your review, click on **Library Home** to return to the documentation home page.

__ 10. Use the **Glossary** to view the output of any term you would like clarification on. As an example, find out the meaning of the word *hypertext*.

- Expand **AIX 5L Version 5.2 Books**
- Expand **Reference Documentation**
- Click on **Glossary**
- **Hint:** AIX V5.1 documentation shows an **Alphabetical List**
- The screen will be divided into two windows. The window on the left is a contents of the Glossary, while the larger screen will display the text output.
- Use the scroll bar on the right if necessary to scroll to find the term *hypertext*. (or use the Netscape menu bar - Edit - Find in Page)
- When you have completed your review, click on **Back**.
- If time permits, feel free to review the meaning of other terms. When you are finished, click on **Library Home** to return to the documentation home page.

__ 11. Use the advanced search function to find information. For example, let's say you wish to find information about the **lpstat** command and the word **printing**.

- Click on **Advanced Search**, which is located towards the upper right of the window.
- Note that you will now be able to type in search strings as well as select which manuals you wish to search in. Click on the first search input window and type in **lpstat**.
- Click on the second search input window and type in **printing**.
- Click on **Search This View.** Wait for the results which appear in a new window
- Click on any output item of interest and review that item. When you are finished, close the window to return to the documentation home page. If time permits, feel free to work further with the search function.

__ 12. From the AIX V5.2 Documentation Library, review what you have done during this session. You will have the ability to immediately re-access any of the documentation windows you have used so far in this session.

- Click on **Go** on the Netscape menu bar. A list will be displayed showing all the documentation windows you have accessed while in this session.
- Click on one of the items in the list to re-access that information.
- When you are finished, click on **Back** or **Library Home** to return to the documentation home page.

__ 13. Exit from the AIX V5.2 Documentation. If you were to continue running in the AIXwindows environment, you could also choose to iconify your documentation window.

- Click on **File** on the Netscape menu bar
- Click on **Exit**
- From AIXwindows, press **<Ctrl><Alt><Backspace>** to return to command line mode.

END OF EXERCISE

Exercise 3. Files and Directories

What This Exercise Is About

This exercise provides the student 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

__ 1. Login to the system.

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

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

__ 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.

__ 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, do the key sequence **<Ctrl-c>** to end the command.)

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

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

__ 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.

__ 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? _____

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

__ 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?

__ 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**.

END OF EXERCISE

Exercise Instructions With Hints

__ 1. Login 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 login.

- $ **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, do 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 the command 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

Solutions

5. **-a** displays all hidden files (files that begin with a **.**). **-R** displays files recursively in a directory structure.

11. The 'Last Accessed' date will be updated anytime the directory is viewed. The other dates are updated when the directory or its i-node structure is changed.

Exercise 4. Using Files

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.

Exercise Instructions

Checking Your Environment

__ 1. Log in to the system.

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

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

Working With Files

__ 4. Look at the contents of the **/etc/motd** and **/etc/passwd** files. Use the commands **cat, pg,** and **more** to see how each command handles the output. The **/etc/motd** file contains the Message of the Day, what you see after you first login. The **/etc/passwd** file contains a list of all the users authorized to use the system.

__ 5. Check the printer queues of your system. Print out the file **/etc/motd**.

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

__ 7. Copy the file **/usr/bin/cal** into your current (home) directory.

__ 8. List the files in your current directory. You should see the two you just copied.

Creating And Manipulating Directories

__ 9. Create a subdirectory in your home directory called **myscripts**.

__ 10. 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.

__ 11. Make the new subdirectory, **myscripts**, your current directory.

__ 12. List the contents of the directory to make sure that the files were copied.

__ 13. Use the mycat command to list the **.profile** file in your home directory.

__ 14. Make your home directory the current directory.

__ 15. Create another subdirectory in your home directory called **goodstuff**.

__ 16. Copy a file called **/etc/profile** into the new directory, and name the new file **newprofile**.

__ 17. Use the **cat** command to look at the file. Hard to read? Try the **pg** command.

__ 18. The file named **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

__ 19. 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.

Remove A Directory

__ 20. Ensure you are in your home directory. Remove the **goodstuff** directory. Could you do it?_____ Why not? _____

__ 21. 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.

END OF EXERCISE

Exercise Instructions With Hints

Checking Your Environment

__ 1. 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 login.

- **$ 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/motd** and **/etc/passwd** files. Use the commands **cat, pg,** and **more** to see how each command handles the output. The **/etc/motd** file contains the Message of the Day, what you see after you first login. The **/etc/passwd** file contains a list of all the users authorized to use the system.

- **$ cat /etc/motd**
- **$ cat /etc/passwd**
- **$ pg /etc/motd**
- **$ pg /etc/passwd**
- **$ more /etc/motd**
- **$ more /etc/passwd**

__ 5. Check the printer queues of your system. Print the file **/etc/motd**.

- **$ qchk -A** (or lpstat)
- **$ qprt /etc/motd**

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

- **$ cp /usr/bin/cat /home/teamxx**

  -OR-

- **$ cp /usr/bin/cat .**

__ 7. Copy the file **/usr/bin/cal** into your current (home) directory.

- **$ cp /usr/bin/cal /home/teamxx**

  -OR-

- **$ cp /usr/bin/cal .**

__ 8. List the files in your current directory. You should see the two you just copied.

- **$ ls**

Creating And Manipulating Directories

__ 9. Create a subdirectory in your home directory called **myscripts**.

- **$ mkdir myscripts**

__ 10. 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**

__ 11. Make the new subdirectory, **myscripts**, your current directory.

- **$ cd myscripts**

__ 12. List the contents of the directory to make sure that the files were copied.

- **$ ls**

__ 13. Use the **mycat** command to list the **.profile** file in your home directory.

- **$ mycat ../.profile**

  -OR-

- **$ mycat /home/teamxx/.profile**

__ 14. Make your home directory the current directory.

- **$ cd**

__ 15. Create another subdirectory in your home directory called **goodstuff**.

- **$ mkdir goodstuff**

__ 16. Copy a file called **/etc/profile** into the new directory, and name the new file **newprofile**.

- **$ cp /etc/profile goodstuff/newprofile**

__ 17. Use the **cat** command to look at the file. Hard to read? Try the **pg** command.

- **$ cat goodstuff/newprofile**
- **$ pg goodstuff/newprofile**

__ 18. The file named **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

- **$ mv goodstuff/newprofile goodstuff/np**
- **$ ls goodstuff**

__ 19. 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

__ 20. Ensure you are in your home directory. Remove the **goodstuff** directory. Could you do it?_____ Why not? _____

- **$ pwd**
- **$ rmdir goodstuff**

__ 21. 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

__ 1. 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**
- **$ pwd**
  /home/teamxx/myscripts
- **$ mkdir -p sports/tennis sports/basketball sports/baseball**
- **$ ls sports**

__ 2. 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**
- **$ cat > sports/basketball/myteam**
  Put anything you want in here
  <ctrl-d>
- **$ cat > sports/basketball/myplayer**
  Put anything you want in here
  <ctrl-d>
- **$ ls sports/tennis sports/basketball**

__ 3. Use the **rm** command to remove the **sports** directory and everything in it.

- **$ pwd**
  /home/teamxx/myscripts
- **$ rm -r sports**

END OF OPTIONAL EXERCISES

Solutions

Following are the solutions for those instructions that include questions:

20. 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 5. File Permissions

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 a student 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

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**. _____

__ 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.

__ 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? _____

__ 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? _____

__ 5. Change 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? _____

__ 6. Remove **home_mycal**. Did that remove **myscripts/mycal**? _____
Why or why not? _____

Working With File Permissions

__ 7. Change 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.

__ 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.

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

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

Working With Directory Permissions

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

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

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

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

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

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

Exercise Instructions With Hints

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 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 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 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 can't 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 don't 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 can't find **mycal** because you don't 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 can't 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 6. vi Editor

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 completing 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.

Exercise Instructions

Creating A File

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

__ 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.

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.

__ 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.

__ 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 instruction 6, 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. Scroll the screen up 1/2 the window size

   iv. Move cursor to last line in the file

   v. Move cursor to first line in the file

   vi. Move cursor to line 4 of the file

   vii. Move cursor to end of line

   viii. Move cursor to beginning of line

__ 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**. Don't forget the space after the word.

__ 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.

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

__ 11. Opening up 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 DON'T exit the editor.

__ 12. While still in command mode, remove the alphabetic characters **c,e,g** but leave the blank lines in their place; in other words, don't 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.

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

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

__ 15. Edit **vitest** one more time. First, copy the first paragraph 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.

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

__ 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.

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

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

__ 20. Write the file and quit the editor.

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.

__ 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**

__ 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.

__ 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!

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

__ 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.

END OF EXERCISE

Exercise Instructions With Hints

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 again (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>** 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 instruction 6, 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. Scroll the screen up 1/2 the window size

iv. Move cursor to last line in the file

v. Move cursor to first line in the file

vi. Move cursor to line 4 of the file

vii. Move cursor to end of line

viii. Move cursor to beginning of line

- **<ctrl-f> or press the Page Down key** (No Page Down key on ASCII terminals)
- **<ctrl-b> or press the Page Up key** (No Page Up key on ASCII terminals)
- **<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**. Don't 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** (lowercase o opens the line after the cursor)

__ 11. Opening up 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 DON'T 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, don't 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 don't 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 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 3yy**
- **<shift-G>; Press p**

__ 16. You just decided that the lines you just added to the end of file don't 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)
- **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

27.

Exercise 7. Shell Basics

What This Exercise Is About

This exercise will familiarize the student 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

Wildcards

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

__ 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? _____

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

__ 4. List all two character file names.

__ 5. List all file names starting with the letters **a, b, c,** or **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 didn't expect? _____ If so, do you know why? _____

__ 7. Return to your home directory.

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.

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

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

Pipes, Tees, And Filters

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

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

__ 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? _____

__ 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.

__ 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 don't. It might not be a straight overwrite because the file is being used twice in the same command. Experiment if you are curious.

__ 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.

2) Do not append the results of the **wc** command to **junk3**. Have the count display to your terminal.

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? _____

__ 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. 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 in **file4** and redirect that number to a file named **file5**.

END OF EXERCISE

Exercise Instructions With Hints

Wildcards

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

- **$ 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 didn't 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.

- **$ cat > junk**

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

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

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

__ 10. Mail the file **junk** to yourself. Wait several seconds 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**

__ 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

don't. 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**

__ 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.

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. 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 in **file4** and redirect that number to a file named **file5**.

- **$ ls -al | tee reverse.listing | wc -w | tee file1 **

  > | tee file2 | tee file3 | tee file4 | wc - l > file5

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 didn't 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 8. Using Shell Variables

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 backquotes look very similar. The single quotes look like this ', and the backquotes like this ‘. The backquote may look different on the keyboard than it does as printed in this exercise.

Exercise Instructions

Variable Substitution

__ 1. Display the shell built-in variables.

__ 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.

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

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

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

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

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

__ 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? _____

__ 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.

__ 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.

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.

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

__ 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**.

Quoting

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

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

__ 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

Exercise Instructions With Hints

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?"** Why is it necessary to use the quotes with **"You Rang?"**? _____ (Single quotes will also work)

- **$ 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?"** Why is it necessary to use the double 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? _____
    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.

    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 logout, 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.

    i. **$ echo '* $n ‘ls‘ $(ls)'**
    _____

    ii. **$ echo "* $n ‘ls‘ $(ls)"**
    _____

    iii. **$ echo * \$n \‘ls\‘ \$\(ls\)**
    _____

    iv. **$ echo * $n ‘ls‘ $(ls)**
    _____

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

    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 backquote.
- f**ilea hello filea filea**
- **filea hello ls**

Exercise 9. Controlling Processes

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 a student 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.

Exercise Instructions

Process Structure

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

__ 2. Create a subshell by entering **ksh.** What is the process id of the subshell? Is it different from your login process?

__ 3. Enter the command **ls -R / > outfile 2> errfile &** and then execute the command which displays all of your running processes. The **ls** command will terminate when it gets to the end of the file system.

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

Process Environment

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

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

__ 7. Create a subshell. 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**?

__ 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**.

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

__ 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?

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

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

__ 13. Create another shell script and name it **sc2**. Have it read:
var1=hello; var2=$LOGNAME; export var1 var2

__ 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?

__ 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?

Job Control

__ 16. Execute the command **ls -R / > outfile 2> errfile** in the foreground.

__ 17. Suspend the job you just started.

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

__ 19. Bring the job back to the foreground.

__ 20. Once the **ls** command finishes executing, restart it again in the background, display the process id, and logoff.

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

__ 22. Create a shell script and name it sc3. It should read:
sleep 60
ls -R / > outfile 2> errfile &
Make it executable. Start the script with the **nohup** command, reference it using an explicit path and put it in the background. Don't forget to redirect the output from **sc3** then logoff.

__ 23. Log in. Check to see if the process is still running.

__ 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**)

Terminating A Process

__ 25. Use the **ls -R /** 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. _____

__ 26. 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.

__ 27. Repeat instruction 25 above. Kill the process using the job number rather than the process id. Check to be sure the job was killed.

END OF EXERCISE

Exercise Instructions With Hints

Process 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 -R / > outfile 2> errfile &** and then execute the command which displays all of your running processes. The **ls** command will terminate when it gets to the end of the file system.

- **$ ls -R / > 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 **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. Execute the command **ls -R / > outfile 2> errfile** in the foreground.

- **$ ls -R / > 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 **ls** command finishes executing, restart it again in the background, display the process id, and logoff.

- **$ ls -R / > outfile 2> errfile &**
- **$ ps -f**
- **$ 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 -f**

__ 22. Create a shell script and name it sc3. It should read:
sleep 60
ls -R / > outfile 2> errfile &
Make it executable. Start the script with the **nohup** command, reference it using an explicit path and put it in the background. Don't forget to redirect the output from **sc3** then logoff.

- **$ vi sc3**
  (press i to insert text)
  sleep 60
  ls -R / > outfile 2> errfile &
  (press the ESC key followed by **:wq** to save the file)
- **chmod 700 sc3**
- **$ nohup ./sc3 > sc3.out 2> sc3err &**
- **$ 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.

- Login: **teamxx**
- teamxx's Password: **teamxx**
- **$ ps -f**

__ 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**

Terminating A Process

__ 25. Use the **ls -R /** 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 -R / > outfile 2> errfile &**

__ 26. 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**

__ 27. Repeat instruction 25 above. Kill the process using the job number rather than the process id. Check to be sure the job was killed.

- **$ ls -R / > 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 logoff 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's 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 **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? _____
    Once you know the process id, kill the process. Check to be sure it was killed.

    Answer: Use **ps -f.**

Exercise 10. Customizing the User Environment

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 Window session, when instructed to log out, execute one of the following commands: **$ su -** or **$ login**

Exercise Instructions

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:

   a) Change the primary prompt string to reflect the current directory.

   b) A message that will be displayed at login which contains your login name and the time you logged in.

   c) Define an alias named **dir** that invokes the **ls -l** command.

   d) Automatically set up the command line editing facility.

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

   a) Did your message display? _____

   b) Is your prompt the name of your home directory? _____

   c) Change to the **/etc** directory. Did your prompt change? _____

   d) Using **dir** do you get a long listing of your current directory? _____

   e) Invoke **dir** using command line editing.
   If you answered NO to any question, edit your **.profile** and fix it.

__ 3. Once you have your customized **.profile** setup and functioning, open a subshell. Answer the following questions:

   a) Is your prompt the name of the current directory? _____

   b) Does the value of the alias dir still work? _____

   c) Can you invoke command line editing? _____

__ 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 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 customization you did in instruction 1.

   Remove JUST what you previously added to the **.profile** in instruction 1, except for the **echo** statement and the **PS1** variable assignment. Add the **ENV** variable assignment. Export both **PS1** and **ENV**.

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

   a) Is your prompt the name of the current directory? _____

   b) Is the value of the alias dir still working? _____

   c) Can you invoke command line editing? _____

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

__ 8. The **dir** alias is still in your **.kshrc** file but isn't 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**.

END OF EXERCISE

Exercise Instructions With Hints

Customizing .profile and .kshrc

__ 1. To customize your environment and have it take effect every time you login, 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:

a) Change the primary prompt string to reflect the current directory.

b) A message that will be displayed at login which contains your login name and the time you logged in.

c) Define an alias named **dir** that invokes the **ls -l** command.

d) 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 logout and back in, or simply rerun it using the dot notation. Once you have done that, execute and answer the following:

a) Did your message display? _____

b) Is your prompt the name of your home directory? _____

c) Change to the **/etc** directory. Did your prompt change? _____

d) Using **dir** do you get a long listing of your current directory? _____

e) Invoke **dir** using command line editing.

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

- **$ logout**
- **Login: teamxx**

  teamxx Password:

  -OR-

- **$ . .profile**

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

- /home/teamxx=> **ksh**

Answer the following questions:

a) Is your prompt the name of the current directory? _____

b) Does the value of the alias dir still work? _____

c) Can you invoke command line editing? _____

__ 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 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 customization you did in instruction 1.

Remove JUST what you previously added to the **.profile** in instruction 1, except for the **echo** statement and the **PS1** variable assignment. Add the **ENV** variable assignment. Export both **PS1** and **ENV**.

- **<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:

a) Is your prompt the name of the current directory? _____

b) Is the value of the alias dir still working? _____

c) 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 isn't 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-

- Logout 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 logout and back in, or simply rerun it using the dot notation. Once you have done that, execute and answer the following:

    a) Did your message display? _____

    b) Is your prompt the name of your home directory? _____

    c) Change to the **/etc** directory. Did your prompt change? _____

    d) Using **dir** do you get a long listing of your current directory? _____

    e) Invoke **dir** using command line editing.

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

    Answers:

    a) A message similar to the following should be displayed:
    User teamxx logged in at Thu Apr 19 14:34:26 CST 2001

    b) Your primary prompt string should be similar to:
    /home/teamxx=>

    c) /home/teamxx=> cd /etc

    d) /etc=> dir
    You should see a long listing of the current directory.

    e) ESC
    k
    /etc=> dir

3. Once you have your customized **.profile** setup and functioning, open a subshell. Answer the following questions:

    a) Is your prompt the name of the current directory? _____

    b) Does the value of the alias dir still work? _____

    c) 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:

a) Is your prompt the name of the current directory? _____

b) Is the value of the alias dir still working? _____

c) 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:

a) Your primary prompt string should be similar to:
/home/teamxx=>

b) /home/teamxx=> dir
You should see a long listing of the current directory.

c) ESC
k
/home/teamxx=> dir

Exercise 11. AIX Utilities (1)

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.

Exercise Instructions

The find Command

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

__ 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.

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

__ 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**.

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

__ 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. _____

END OF EXERCISE

Exercise Instructions With Hints

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. _____

a) **$ cd**

b) **$ mkdir level1**

c) **$ touch level1/letter1**

d) **$ cd level1**

e) **$ mkdir level2**

f) **$ touch level2/letter2**

g) **$ cd**

h) **$ ls l***

i) **$ 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 find only 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 12. AIX Utilities (2)

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.
- Login 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.

Exercise Instructions

The grep Command

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

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

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

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

__ 5. Use the **ps** and **grep** commands to display the processes initiated by users other than yourself.

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.

The head And tail Command

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

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

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

The tn, ftp and tar Commands

__ 11. Login 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.

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

__ 13. On your remote system change to the **/tmp**-directory and create a new file **testfile1**.

__ 14. Logout from the remote system.

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

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

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

__ 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.

__ 19. Verify the content of the archive file.

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

END OF EXERCISE

Exercise Instructions With Hints

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.

- **$ ps -ef | grep -v teamxx**
  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. Login 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. Logout 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 /tmp/archive.tar**

END OF EXERCISE

Exercise 13. AIX Utilities (3)

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

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**.

__ 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**, etc. Verify that the files were copied and the names assigned accordingly.

__ 3. Copy the **/etc/passwd** file into your **newdir** directory so that you will have one file in the directory containing text. All the other files in the directory will be empty. Using **find, xargs,** and **grep**, **cat** the contents of the files that contain text.

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

__ 5. Using the **find** command, create a "recursive" listing of all files starting in your home directory and redirect the output to a file named **myfiles**. Using the redirected file, **myfiles**, determine the type of files located in the list.

END OF EXERCISE

Exercise Instructions With Hints

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**, etc. Verify that the files were copied and the names assigned accordingly.

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

__ 3. Copy the **/etc/passwd** file into your **newdir** directory so that you will have one file in the directory containing text. All the other files in the directory will be empty. Using **find, xargs,** and **grep**, **cat** the contents of the files that contain text.

- **$ cp /etc/passwd .**
- **$ find . -type f | xargs -t grep ".*" | cat**

__ 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**

__ 5. Using the **find** command, create a "recursive" listing of all files starting in your home directory and redirect the output to a file named **myfiles**. Using the redirected file, **myfiles**, determine the type of files located in the list.

- **$ ls -R > myfiles**
- **$ file -f myfiles | pg**

  -OR-
- **$ find $HOME > myfiles; file -f myfiles**

  -OR-
- **$ find $HOME | xargs file**

END OF EXERCISE

Exercise 14. AIX Utilities (4)

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

Using diff, cmp, dircmp

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

   a) Change the spelling of one of the names

   b) Remove one of the names

   c) Add a new name

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

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

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

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. _____

__ 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.

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

__ 8. Using **uncompress**, restore the compress 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 instruction 10.

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, etc., between the words. Display the file.

__ 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, etc.

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

__ 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.

a) Invoke a listing of files and directories in your home directory (**invisdir** should be included as part of the output)

b) Try to invoke a long listing on the **invisdir** directory.

c) Try to remove the **invisdir** directory.

d) Repeat instruction a. above to see if there are any non-printable characters in the **invisdir** directory name that made instructions b. and c. fail.

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

END OF EXERCISE

Exercise Instructions With Hints

Using diff, cmp, dircmp

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

a) Change the spelling of one of the names

b) Remove one of the names

c) 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 compress 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 instruction 10.

- **$ 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, etc. 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, etc.

- **$ 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.

a) Invoke a listing of files and directories in your home directory (**invisdir** should be included as part of the output)

b) Try to invoke a long listing on the **invisdir** directory.

c) Try to remove the **invisdir** directory.

d) Repeat instruction a. above to see if there are any non-printable characters in the **invisdir** directory name that made instructions b. and c. 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 15. Additional Shell Features

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

Positional Parameters And Conditional Execution

__ 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.

__ 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.

__ 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? _____

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

__ 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**.

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.

__ 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.)

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.

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

END OF EXERCISE

Exercise Instructions With Hints

__ 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:

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.

   Answers:
   The name of this shell script is parameters.
   The first parameter passed is number 10.
   The second parameter passed is number 100.
   The third parameter pass is number 1000.
   Altogether there were 3 parameters passed.

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 16. Using AIXwindows

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
- (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 from a graphics terminal. 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.

Exercise Instructions

Working with windows

__ 1. Log in to your system

__ 2. Start AIXwindows using the **startx** command. Note how the windows are displayed first, then Motif adds its frames around the windows. What windows are displayed?

__ 3. Verify that the *aixterm* is the active window.

__ 4. Using the *aixterm*, try typing some AIX commands such as **ls, date, cal, whoami**.

__ 5. Resize the width of the window.

__ 6. Change the height and width of the window, simultaneously.

__ 7. Drag the *aixterm* from one side of the screen to the other.

__ 8. Use the options in the window menu to move and resize the window.

__ 9. View the window menu again. Why do you think some items may be "greyed out"?

__ 10. Open the window menu on the *aixterm*, but now type the letter "m" rather than clicking on *move*. Note that this is another way to move a window.

Try the mnemonics for some of the other functions.

__ 11. 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?

__ 12. Iconify (minimize) the *aixterm* window. Once it is an icon, restore it back to the screen.

__ 13. Maximize the *aixterm* window. What happens? Once it is maximized, resize the window to a smaller size.

Using the root window

__ 14. Use the root menu to open another *aixterm* window.

__ 15. Start another *xclock* from the root menu.

Command line options for aixterm

__ 16. 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.

__ 17. Start an *aixterm* from the command line. Give the window the following characteristics:

background color	lightskyblue
foreground color	forestgreen
font	rom14.iso1
title	My Window
full cursor	
scrollbar	

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

__ 18. 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

Cut, paste, and reexecute functions

__ 19. 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.

__ 20. Within a second *aixterm* window, use the vi editor to create another file called *tempfile.new*. Go to 'insert' mode but do not add any text to this file at this time.

__ 21. 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.

__ 22. You have now completed this machine exercise. You may either try the optional machine exercise, end AIXwindows or lock your terminal.

The client-server model (optional)

__ 23. Use the **xhost** command to enable all other clients to access your X server.

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

__ 24. Check to see if the DISPLAY variable has been set. If not, set it. Before you can set it you will need to know the TCP/IP name of your computer. This information can be obtained using the **hostname** command.

__ 25. Start an *aixterm* session on someone else's computer, but have it display on your system. You will need to know the other system's **hostname** to do this. When asked for a login ID use **remote1**. A new window should appear.

__ 26. In the new window you just started, use the **hostname** command to verify that the window is running from the remote system. Check the value of the DISPLAY variable. It should indicate the name of your host.

__ 27. 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.

__ 28. You have now completed the optional machine exercise. Either close AIXwindows or lock your terminal.

END OF EXERCISE

Exercise Instructions With Hints

Working with windows

__ 1. Log in to your system

- login: **teamxx**
- password **<your password>**

__ 2. Start AIXwindows using the **startx** command. Note how the windows are displayed first, then Motif adds its frames around the windows. What windows are displayed?

- **$ startx**

__ 3. Verify that the *aixterm* is the active window.

- Move the mouse pointer to the *aixterm* and click with the left mouse button. The window frame will change color to indicate that this is the active window.

__ 4. Using the *aixterm*, try typing some AIX commands such as **ls, date, cal, whoami**.

__ 5. Resize the width of the window.

- Move the mouse pointer to the window frame on the right hand 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.

__ 6. 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.

__ 7. 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.

__ 8. 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. They should perform the same functions as above.
- 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 when you have finished resizing.

__ 9. View the window menu again. Why do you think some items may be "greyed out"?

__ 10. Open the window menu on the *aixterm*, but now type the letter "m" rather than clicking on *move*. Note that this is another way to move a window.

- Click on 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.

__ 11. 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?

__ 12. Iconify (minimize) the *aixterm* window. Once it is an icon, restore it back to the screen.

- Use the left mouse button to click on 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 on the icon with the left mouse button. The window menu is again displayed, with certain options activated. Using the left mouse button, click on "restore". This will restore the window with its previous location and size. Double clicking on an icon will also restore that window.

__ 13. Maximize the *aixterm* window. What happens? Once it is maximized, resize the window to a smaller size.

- Using the left mouse button, click on 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

__ 14. 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** will be displayed.

__ 15. 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.

Command line options for aixterm

__ 16. 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**

__ 17. 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?

__ 18. 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 &**

Cut, paste, and reexecute functions

__ 19. 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

__ 20. 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

__ 21. 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 press the center mouse button. The text will be placed beginning at the cursor location.
- **<Esc>** and :wq

__ 22. You have now completed this machine exercise. You may either try the optional machine exercise, end AIXwindows or lock your terminal.

- Continue with the optional machine exercise
  -OR-
- **<Ctrl><Alt><Backspace>** to close AIXwindows
  -OR-
- Move the mouse pointer to the root menu and press the right mouse button.
- On the root menu choose "Clients".
- From the "Client" menu choose "Screen Lock".
- To unlock the screen, type your AIX user password.

The client-server model (optional)

__ 23. 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".

__ 24. Check to see if the DISPLAY variable has been set. If not, set it. Before you can set it you will need to know the TCP/IP name of your computer. This information can be obtained using the **hostname** command.

- **hostname**
- **echo $DISPLAY**
- If no value has been set for DISPLAY, set it now:
  DISPLAY=<hostname>:0 (Do not use the '<>' in the command)

__ 25. Start an *aixterm* session on someone else's computer, but have it display on your system. You will need to know the other system's **hostname** to do this. When asked for a login ID use **remote1**. A new window should appear.

- **rexec other_host aixterm -display $DISPLAY**

__ 26. In the new window you just started, use the **hostname** command to verify that the window is running from the remote system. Check the value of the DISPLAY variable. It should indicate the name of your host.

- **hostname** (should show remote system)
- **echo $DISPLAY** (should show your system name)

__ 27. 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**

__ 28. You have now completed the optional machine exercise. Either close AIXwindows or lock your terminal.

- **<Ctrl><Alt><Backspace>** to end AIXwindows
  -OR-
- Move the right mouse pointer to the root window and press the right mouse button.
- On the root menu, choose "Clients".
- From the "Clients" menu choose "Screen Lock".
- To unlock the screen, type in your AIX user password.

END OF EXERCISE

Solutions

2. Start AIXwindows using the **startx** command. Note how the windows are displayed first, then Motif adds its frames around the windows. What windows are displayed?

   Answer: One *aixterm* and one *xclock* are displayed.

9. 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.

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

    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.

17. 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

27. From the remote systems window, execute the **xcalc &** command. From which system is the calculator being executed? You can verify this with the **ps** command.

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

Exercise 17. Customizing AIXwindows (1)

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

***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.)

__ 1. Log in to your AIX system.

__ 2. Copy the file **/usr/lpp/X11/defaults/xinitrc** into your $HOME directory and call the file **.xinitrc.**

__ 3. Edit the .xinitrc file and make the following changes:

Add a second hand to the xclock

Make the root window solid black

Start another aixterm with a title like "Bills Window"

__ 4. Start AIXwindows. Does the AIXwindows environment look different? It should!

Customizing the .Xdefaults file

__ 5. Execute the command **aixterm -keywords | pg** to view all the resources that can be customized for an *aixterm* window.

__ 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

__ 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?

__ 8. Now, end AIXwindows and then restart AIXwindows. What do the two original windows look like? Why?

__ 9. Edit the **.Xdefaults** file and update the following lines for new colors:

Aixterm*foreground: grey

Aixterm*background: navy

__ 10. Restart the mwm and then create a new *aixterm* window from the command line. Does it use your new color specifications? It should!

__ 11. Exit your AIXwindows environment and logout from the system.

END OF EXERCISE

Exercise Instructions With Hints

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.)

__ 1. Log in to your AIX system.

- login: **teamxx**
- passwd: **<your password>**

__ 2. Copy the file **/usr/lpp/X11/defaults/xinitrc** into your $HOME directory and call the file **.xinitrc.**

- **cd**
- **cp /usr/lpp/X11/defaults/xinitrc .xinitrc**

__ 3. Edit the .xinitrc file and make the following changes:

Add a second hand to the xclock

Make the root window solid black

Start another aixterm with a title like "Bills Window"

- **vi .xinitrc**
- 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 AIXwindows. Does the AIXwindows environment look different? It should!

- **startx**

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 on **OK**.
- Using the left mouse button, click on 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 AIXwindows and then restart AIXwindows. What do the two original windows look like? Why?

- **<Ctrl> <Alt> <Backspace>** to end AIXwindows
- $ **startx**

__ 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 on **OK**.
- Using the left mouse button, click on one of your *aixterm* windows so that it becomes the active window.
- On the command line enter: **aixterm &**

__ 11. Exit your AIXwindows environment and logout from your system.

- Press <CTRL><ALT><BACKSPACE>.

END OF EXERCISE

Exercise 18. Customizing AIXwindows (2)

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 scroll bar
- 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

Using the Custom Tool: Color and Fonts

__ 1. Log in to your system and start AIXwindows.

__ 2. Make sure you have two *aixterm* windows open as well as the *xclock*. Also, start the scientific calculator.

__ 3. Start the AIXwindows customization tool.

__ 4. On the Customizing Tool window, choose *xcalc*.

__ 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.

__ 6. Change the background color for *xcalc* to the color of your choice.

__ 7. Switch focus to an *aixterm* window and display the contents of **.Xdefaults.** Has it been updated? It shouldn't have been!

__ 8. So, to have your values saved in **.Xdefaults**, change your focus back to the *xcalc* customizing window. Save the values you have chosen.

__ 9. Now, review the **.Xdefaults** file again. Your resource change should now be there.

__ 10. Return to the *xcalc* Customizing window and now choose the resource category of **Fonts**.

__ 11. View the various fonts that can be used for the *window interior*.

__ 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.

__ 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?

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.

__ 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.

__ 16. Have the root window display snowflakes or escherknots - take your pick. These bitmap images are found in the directory **/usr/include/X11/bitmaps**. You may wish to view the filenames 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.

__ 17. If you decide you like any of these root window options, how would you make your customization permanent, i.e. available every time you start AIXwindows?

Optional Exercises

Using the Custom Tool: Size and Location, Icons and Scroll Bar

__ 1. Start the AIXwindows Custom Tool and choose *xcalc* again.

__ 2. Choose the Size and Location resource category and change the size of the *xcalc*.

__ 3. 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.

__ 4. 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.

__ 5. 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.

Customizing the Motif Window Manager (mwm)

__ 6. 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.

__ 7. Some users prefer to use the pointer focus policy so they do not have to click on 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.

END OF EXERCISE

Exercise Instructions With Hints

Using the Custom Tool: Color and Fonts

__ 1. Log in to your system and start AIXwindows.

- **$ startx**

__ 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 on **xcalc**. The line should be highlighted.
- Use the left mouse button to click on **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 on 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 on **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 on 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 on 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 on Match RGB to Closest Color Name, and review the results.

- Click on **Apply**. The background color of the *xcalc* should change.
- Click on **OK** to close the window.

__ 7. Switch focus to an *aixterm* window and display the contents of **.Xdefaults**. Has it been updated? It shouldn't have been!

- Use the left mouse button to click on 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 on **File**, which is located in the upper left of the window.
- Click on **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 on **OK**

__ 9. Now, review the **.Xdefaults** file again. Your resource change should now be there.

- Use the left mouse button to click on 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 on the Customizing Tool window.
- Click on 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 on 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 on 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.

- 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've decided on the font to use, be sure it is highlighted and then click on **Apply**. *xcalc* should now use the new font.
- Click on **OK**.
- On the *xcalc* customizing window, click on **File**
- Click on **Save As**...
- Click on **OK** to save your changes to the **.Xdefaults** file.
- Click on the upper left of the *xcalc* Customizing Tool window to open the window menu and then click on **Close**.
- **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?

- 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 on **aixterm** and then on **OK**.
- Click on the **Colors**... box for window interior background.
- On the Colors window, choose any color and then click on **OK**
- On the *aixterm* Customizing window, click on **File**. Then, click on **Save as**.... On the Save As... window, click on **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 on **Close**.

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**
- **xsetroot -cursor_name shuttle**
- **xsetroot -cursor_name gumby**
- Move the cursor to the root window to view the new cursor shape.

__ 16. Have the root window display snowflakes or escherknots - take your pick. These bitmap images are found in the directory **/usr/include/X11/bitmaps**. You may wish to view the filenames 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, i.e. available every time you start AIXwindows?

- Change the **xsetroot** command in .xinitrc

Optional Exercises

Using the Custom Tool: Size and Location, Icons and Scroll Bar

__ 1. 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 on **xcalc** and then on **OK**.

__ 2. 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 on the small rectangle in the Colors box. Click on **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. The *xcalc* window should change size immediately.
- Save the size values if you want by clicking on **File**. Then, click on **Save as**.... On the Save As... window, click on OK. Your changes have now been added to the **.Xdefaults** file.

__ 3. 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 on the small rectangle in the Colors box and then click on **Icon**.

__ 4. 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 on **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 on the file name and then on **View Picture**. The escherknot is an interesting icon to view.
- Under Files, click on terminal and then **View Picture**. Once you approve of this choice, choose Cancel to remove the picture.
- Click on **OK** to save your choice

- On the *xcalc* Customizing window, use the left mouse button to click on **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.

__ 5. 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 on **aixterm** and then on **OK**.
- Click on the small rectangle in the Colors box and then click on **Scroll Bar**.
- Click on the box for **visible scroll bar** and choose **true**.
- Click on **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 scroll bar resource should be listed.
- Start an *aixterm* window using the **aixterm &** command. The new window should display a scroll bar.

Customizing the Motif Window Manager (mwm)

__ 6. 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 on **Custom**.
- On the Customizing Tool window click on **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 on **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 on 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!

__ 7. Some users prefer to use the pointer focus policy so they don't have to click on 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 on the small rectangle in the Colors box. Then, click on **Focus**
- Move the pointer to the box that corresponds to the keyboard focus policy and click on the small rectangle.
- The default is explicit, meaning that you need to click on a window to make it the active window. If you wish to change the focus policy, click on pointer.
- To save this change, click on **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 on **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

Exercise 19. Using the Common Desktop Environment (CDE)

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.

Exercise Instructions

Exploring the Front Panel

__ 1. Log in as **teamxx**

__ 2. Find these components of the CDE Front Panel:

Workspace Switch Buttons

Style Manager

File Manager

Clock, Calendar

Application Manager

Personal Application Manager

Mail

Trash Can

Exit Icon

Move Handles

Menu and Iconify Buttons

__ 3. Move the Front Panel to the top of the screen.

__ 4. Iconify the Front Panel and then restore it.

Work with the Help Manager

__ 5. Use the Help Manager to learn about Basic Desktop Skills.

__ 6. Use the Help Facilities to Search for Help on using the File Manager to copy files. Follow the links for that item to see what kind of help structure is provided.

__ 7. From the Help subpanel note how options exist so that you can access AIX online documentation. When you have reviewed the various Help functions, close the Help windows.

Starting a Terminal Window

__ 8. Start an *aixterm* Terminal Window.

Now you have a terminal window where commands can be entered.

__ 9. Run some command line commands.

__ 10. Start the Desktop Terminal dtterm, using the Personal Apps front panel popup menu.

__ 11. Run some command line commands.

__ 12. Compare the *aixterm* and the *dtterm* windows. What differences do you see?

__ 13. Now close all open windows, except the Front Panel, and we will work with the File Manager.

Working with the File Manager

__ 14. Select the File Manager control from the Front Panel to access the File Manager.

__ 15. Make sure that you are in your Home Directory - called **/home/teamxx**.

__ 16. So that you have a few items to work with, the first thing you'll need to do is create a few new files.

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.

__ 17. View several of the files.

__ 18. Click on a file name and then click on **Selected** in the menu bar. Click on **Open** to edit the file.

This will invoke the CDE Text Editor.

__ 19. Enter several lines of text in each file. 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 on the **File** option in the menu bar, then selecting **Close**. Confirm that you want to save the file when that window is presented.

__ 20. When you have edited all of the files, pick one of them that is a shell script and add execute to its file permissions. Once this is complete, execute the shell script.

The icon will change to a lightening bolt indicating this file can be executed.

__ 21. To execute the Shell Script, be sure its icon shows as a lightening bolt. Double click on the Shell Script icon. On the Action:Run window click on 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 - among other things - some drag and drop operations.

Drag and Drop operations

You'll need to be working with the files in your Home directory, so these should be displayed in the File Manager window.

If you aren't at the correct directory, navigate up and/or down the structure until you get to where you want/ought/need to be.

__ 22. 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.

__ 23. Drop the file on your workspace backdrop.

The file icon has been dropped onto the backdrop and will stay there for fast, convenient access. Now, if the file is executable, use the left mouse button and double click on the file icon to make it run.

__ 24. 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?

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.

__ 25. While dragging a file, take it across the controls on the desktop.

What do you see?

__ 26. Drop the file on the Clock control. What happens?

The Desktop File Manager is one of the most useful and powerful tools in CDE. This section explores more of the File Manager capabilities.

__ 27. Set the File Manager preferences to display a Directory Tree diagram, starting at the root directory.

__ 28. Expand the **/usr/dt** directory.

__ 29. Set the preferences to display Names and Small Icons, rather than a Tree Structure.

__ 30. Set the preferences to display By Properties. This output will look similar to the output of the **ls -ls** command.

__ 31. Close the File Manager and any windows that it opened.

__ 32. Use the File Manager to execute the **date** command. This command is found in the **/bin** directory.

__ 33. Use the File Manager to create the directory **cdelab** in your $HOME directory.

__ 34. 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.

__ 35. Copy two or more of the pixmap files to the **cdelab** subdirectory.

__ 36. Rename one of the files to **myicon.pm.**

__ 37. Delete the **myicon.pm** file using the mouse and the Front Panel trash can.

__ 38. Delete a second pixmap file using the File Manager Menu Bar

__ 39. With the CDE it is possible to retrieve a deleted file. Restore **myicon.pm.**

__ 40. Empty the trash can.

__ 41. Change the Owner and Group permissions of the restored file to read/write.

__ 42. Close the File Manager

__ 43. At this point you may continue with the optional exercise or exit out of CDE. Skip this step to perform the optional exercise.

END OF EXERCISE

Optional Exercises

__ 1. Click on the Calendar control on the Front Panel. Add an appointment for next week.

__ 2. Change the view to **Day View** to view the appointment you have scheduled.

__ 3. Set a reminder to yourself for the appointment.

__ 4. Change the view to **Week View** to view the appointment you have scheduled. Make the appointment private so that others cannot view it on your calendar.

__ 5. Return to the **month view** icon on the calendar menu bar.

__ 6. Close the Calendar window and exit out of CDE.

END OF EXERCISE

Exercise Instructions With Hints

Exploring the Front Panel

__ 1. Log in as **teamxx**

- Enter your userid into the Login Manager panel and press Enter.
- Enter your password and press Enter.

__ 2. Find these components of the CDE Front Panel:

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

__ 3. Move the Front Panel to the top of the screen.

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

__ 4. Iconify the Front Panel and then restore it.

- To iconify, click on the top right corner of the panel
- To restore the Front Panel: click on the icon and select **Restore**

Work with the Help Manager

__ 5. Use the Help Manager to learn about Basic Desktop Skills.

- Click on the arrow above the Help Manager
- Select the **Help Manager**
- Review the Help that you get for the Desktop
- Select the **Overview and Basic Desktop Skills** volume and follow some of the hyperlinks

__ 6. Use the Help Facilities to Search for Help on using the File Manager to copy files. Follow the links for that item to see what kind of help structure is provided.

- Select the **Search** option on the Help Viewer menu bar
- Select **Index...** on the menu that drops down
- Select **All Volumes** and wait for the results
- Click **File Manager Help** in the index list
- Scroll down to **copying**, **file** and click
- The text will be displayed in the **Help Viewer** window

__ 7. From the Help subpanel note how options exist so that you can access AIX online documentation. When you have reviewed the various Help functions, close the Help windows.

Starting a Terminal Window

__ 8. Start an *aixterm* Terminal Window.

- Select the **Application Manager** control (the file cabinet icon next to Help)
- Double click on the **Desktop Tools** icon with the left button of the mouse to display the tools available. One of them is Aixterm (that will use your .Xdefaults file).
- Double click the **Aixterm** icon with the left mouse button.

Now you have a terminal window where commands can be entered.

__ 9. Run some command line commands.

- Enter one or more of these: **xcalc &** ; **ls** ; **ps**

__ 10. Start the Desktop Terminal dtterm, using the Personal Applications Front Panel popup menu (the control that looks like a piece of paper and a pencil).

- Click on 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 on the **Terminal** menu item. This will display a *dtterm*.

__ 11. Run some command line commands.

- Enter one or more of these: **xcalc &** ; **ls** ; **ps**

__ 12. Compare the *aixterm* and the *dtterm* windows. What differences do you see?

- *ddterm* has a window menu and scroll bar.
- In *dtterm*, use the Edit menu bar option to copy and paste text.
- 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.

__ 13. Now close all open windows, except the Front Panel, and we will work with the File Manager.

Working with the File Manager

__ 14. Select the File Manager control from the Front Panel to access the File Manager.

- Click on 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.

__ 15. 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'll 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 on 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 on 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.

__ 16. 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 on the **File** menu bar option. Then, click on **New File**
- Type in the name of the new file in the window presented and click on **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.

__ 17. 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.

__ 18. Click on a file name and then click on **Selected** in the menu bar. Click on **Open** to edit the file. This will invoke the CDE Text Editor.

__ 19. Enter several lines of text in each file. 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 on 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.

__ 20. When you have edited all of the files, pick one of them that is a shell script and add execute to its file permissions. Once this is complete, execute the shell script.

- Click on the file's icon.
- Press the right mouse button.
- Choose the Change Permissions option from the pull down menu displayed. You can see what the current properties are - including File permissions.
- Click on Execute permission for the owner. Click on **OK** to commit the changes.

The icon will change to a lightening bolt indicating the file can be executed.

__ 21. To execute the Shell Script, be sure its icon shows as a lightening bolt. Double click on the Shell Script icon. On the Action:Run window click on **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 - among other things - some drag and drop operations.

Drag and Drop operations

You'll need to be working with the files in your Home directory, so these should be displayed in the File Manager window.

If you aren't at the correct directory, navigate up and/or down the structure until you get to where you want/ought/need to be.

__ 22. 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.

- Select a file by clicking on 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.

__ 23. Drop the file on your workspace backdrop.

The file icon has been dropped onto the backdrop and will stay there for fast, convenient access. Now, if the file is executable, use the left mouse button and double click on the file icon to make it run.

__ 24. 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.

__ 25. 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 - e.g. the Style Manager or the Workspace switches.

__ 26. Drop the file on the Clock control. What happens?

- If you try to drop an item to a control that won't 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.

__ 27. Set the File Manager preferences to display a Directory Tree diagram, starting at the root directory.

- On the File Manager Window, double click on the root directory icon
- Click on **View**
- Click on **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 on **Apply**

__ 28. Expand the **/usr/dt** directory.

- Click on the + in front of the **/usr** icon (you may have to scroll).
- Click on the + in front of the **dt** icon (again, you may have to scroll to see the filenames listed)
- The **/usr/dt** directory contains CDE executables and default configurations.

__ 29. Set the preferences to display Names and Small Icons, rather than Tree Structure.

- 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**

__ 30. Set the preferences to display By Properties. This output will look similar to the output of the **ls -la** command.

- In the Set View window:

  Unselect all choices in Headers
  Select **By Tree** and **Folders Only** in Show;
  Select **by Name, date, size...** in Representation.
  Click **Apply**

__ 31. Close the File Manager and any windows that it opened.

__ 32. Use the File Manager to execute the **date** command. This command is found in the **/bin** directory.

- Click on the **File Manager** icon
- Double click on the root directory icon
- Double click on the bin directory icon
- Scroll and find **date**
- Double click on **date**
- From the Action:Run panel, click on OK
- After **date** executes, close the Run Panel
- Close the File Manager

__ 33. Use the File Manager to create the directory **cdelab** in your $HOME directory.

- Click on the **File Manager** icon
- Verify that the icon path shows **/home/teamxx** (If not, click on **File** and choose **Go Home**)
- Click on **File**
- Click on **New Folder**... (A folder is like a directory to the CDE File Manager)
- Enter New Folder Name: **cdelab**
- Click **OK**
- Close the File Manager
- 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 on the **File Manager** icon
- Click on **File**
- Click on **Find**
- Enter File or Folder Name: ***.pm**
- Enter Search Folder: **/usr/dt**
- Click on **Start**

__ 34. Copy two or more of the pixmap files to the **cdelab** subdirectory.

- Select a **.pm** file by clicking it
- Click on **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

__ 35. Rename one of the files to **myicon.pm.**

- Double click on the **cdelab** icon
- Click on 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**

__ 36. 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

__ 37. Delete a second pixmap file using the File Manager Menu Bar

- Click on a pixmap file icon in your directory
- Click on **Selected**
- Click on **Put In Trash**
- Click **OK** on the Trash Can Warning

__ 38. 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 on the Trash Can icon. This displays the Trash Can window.
- Click on the **myicon.pm** icon
- Click on **File**
- Choose **Put Back** to retrieve a deleted file

__ 39. Empty the trash can.

- Select **File** on the Trash Can window
- Select **Select All**
- Select **File**
- Select **Shred**
- Click on **OK** when the shred warning is displayed. The files will be deleted.
- Close the Trash Can window

__ 40. Change the Owner and Group permissions of the restored file to read/write.

- Click on the **myicon.pm** icon in the File Manager window.
- Click on **Selected**

- Click on **Change Permissions**
- Click on Write for Owner
- Click on Write for Group
- Click on **OK**

__ 41. Close the File Manager

__ 42. At this point you may continue with the optional exercise or exit out of CDE. Skip this step to perform the optional exercise.

- Click on **Exit** on the Front Panel
- Click on **Continue Logout** on the confirmation panel

END OF EXERCISE

Optional Exercises

__ 1. Click on the Calendar control on the Front Panel. Add an appointment for next week.

- Click on the Calendar control on the Front Panel - it is just to the right of the clock
- Click on the Appointment Editor on the Calendar Tool Bar. 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 on the rectangle button to view the various start times. Click on a start time. Click on **AM** or **PM** as necessary.
- In the **End** time window, click on the rectangle button to view the various end times. Click on an end time. Click on **AM** or **PM** as necessary.
- In the **What** window type what the appointment is for.
- Click on **Insert**.
- Click on **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 on the **>** to the right of **Today** on the Calendar Tool Bar to view the next month.

__ 2. Change the view to **Day View** to view the appointment you have scheduled.

- Use the left mouse button to click on the day (on the month view calendar) that the appointment has been scheduled.
- Click on the **Day View** icon on the Calendar Tool Bar. The **Day View** icon is the fourth icon from the right.

__ 3. Change the view to **Week View** to view the appointment you have scheduled.

- Click on the **Week View** icon on the Calendar Tool bar. The **Week View** icon is the third icon from the right.

__ 4. Set a reminder to yourself for the appointment. Make the appointment private so that others cannot view it on your calendar.

- Click on the Appointment Editor icon. Again, this icon is the first icon on the Calendar Tool Bar.
- Be sure your appointment shows in the **Time What** window. If it does not, cancel the Appointment Editor and then click on the day your appointment is scheduled. Once this day is highlighted, click on the Appointment Editor.
- On the Appointment Editor, click on **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 on Privacy and choose a preferred privacy option.
- Once the reminder is complete, click on **Insert**. Then click on **Cancel**.

__ 5. Return to the **month view** icon on the calendar menu bar.

- Click on the **Month View** icon on the calendar menu bar. It is the second from the right.

__ 6. Close the Calendar window and exit out of CDE.

- Click the upper left of the window to display the menu window. Click on **Close**.
- Click on **EXIT** on the Front Panel. Click on **Continue Logout** when asked to confirm.

END OF EXERCISE

Exercise 20. Customizing CDE

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

Customizing the Front Panel

__ 1. Log in as **teamxx**

__ 2. Customize your Workspaces as follows:

Rename each Workspace.

Change the Backdrop of each Workspace.

Turn on the screen saver and screen lock.

Set the window behavior

Select a different palette for the workspaces.

__ 3. Add a fifth workspace and customize its style using the Style Manager.

__ 4. Set the new session as your Home session, and set Startup to return to your Home session at login.

__ 5. Log out and log in again. Check to see that the state of your session matches what you set in the previous steps.

__ 6. Add the same **dtterm** session to all workspaces.

__ 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 an application from a workspace.

__ 9. Have the **dtterm** application appear on the front panel as the default application associated with the Personal Apps control.

__ 10. Tear off the Personal Apps subpanel menu, and place it on the workspace.

__ 11. Create a new subpanel for the Style Manager control and add the Icon Editor and the **aixterm** applications to it.

__ 12. Now, remove the Icon Editor from the new subpanel.

Adding a New Control to the Front Panel

__ 13. Start the **Application Manager**.

__ 14. Open the **Personal Applications** subpanel.

__ 15. Drag the **Netscape Navigator** icon (Application Manager window) onto the **Install Icon** from the Personal Applications subpanel.

If you find no Netscape Navigator icon, choose another icon from the Application Manager.

__ 16. Close the **Personal Applications** subpanel.

__ 17. Find out the name of the definition file in directory $HOME/.dt/types/fp_dynamic. Write down the file name:

_____

__ 18. Copy this definition file to directory **$HOME/.dt/types** and specify a new file name.

__ 19. Anchor the netscape control in the front panel by editing the copied definition file. Use your student notes to find out which lines must be changed.

__ 20. Restart the CDE. After restarting CDE, you should see the **Netscape Navigator** icon on the front panel.

END OF EXERCISE

Exercise Instructions With Hints

Customizing the Front Panel

__ 1. Log in as **teamxx**

__ 2. Customize your Workspaces as follows:

Rename each Workspace.

- Double click on a 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 on the **Style Manager** icon
- Click on the **Backdrop** icon
- Select your choice of backdrop and click on **Apply**
- Click on **Close**
- Select another Workspace and repeat

Turn on the screen saver and screen lock.

- While in the Style Manager, click on the **Screen** icon
- Make your choices and click **OK**

Set the window behavior

- While in the Style Manager, click on the **Window** icon
- Make your choices and click on **OK**

Select a different palette for the workspaces.

- In the Style Manager, click on the **Color** icon
- Select a palette and click on **OK**

__ 3. Add a fifth workspace and customize its style using the Style Manager.

- Click on 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.

- Click on the new Workspace Button.
- In the Style Manager, select the **StartUp** icon.
- Select **Set Home Session...**

- Click on **OK** to replace
- Select **At login, Return to Home Session**
- Click on **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 on **Exit**
- Click on **Continue logout** when asked to confirm

__ 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 on **Terminal**
- Click on the **dtterm**'s window menu button at the top left of the window frame (the dash).
- Note the options on the window menu. Click on **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 an application from a workspace.

- If you want to remove an application 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 on 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 on 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 on 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 on 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 on 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 scroll bar 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 on the **Application Manager** control.
- Click on **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 **Netscape Navigator** icon (Application Manager window) onto the **Install Icon** from the Personal Applications subpanel.

If you find no Netscape Navigator icon, choose another icon from the Application Manager.

- Select the Netscape Navigator icon, and drag it with the left mouse button onto the Install Icon.

__ 16. Close the **Personal Applications** subpanel.

- Click on 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 Netscape1.fp

__ 18. Copy this definition file to directory **$HOME/.dt/types** and specify a new file name.

- **$ cp $HOME/.dt/types/fp_dynamic/Netscape1.fp $HOME/.dt/types/browser.fp**

__ 19. Anchor the netscape control in the front panel by editing the copied definition file. Use your student notes to find out which lines must be changed.

- **$ vi $HOME/.dt/types/browser.fp**

```
CONTAINER_TYPE    BOX
CONTAINER_NAME    Top
POSITION_HINTS    last
```

__ 20. Restart the CDE. After restarting CDE, you should see the **Netscape Navigator** icon on the front panel.

END OF EXERCISE