U1010

UNIX I

August 15, 1996
Training Manual

Author: Johanes Suhardjo
Editor: Cynthia Maciejczyk
Reviewers: Paul Go, Doug McKeown, Milind Saraph, Rich Sudlow

Visit the OIT Documentation Web Page at:
http://www.nd.edu/~doc

To submit comments regarding this document,
fill out the Documentation Comments form at
http://www.nd.edu/~doc/comments.html
or send e-mail to **doc.1@nd.edu**

Introduction

About This Document

The content of this document is elementary; a user without any prior experience with computers will be able to understand it. To learn more about UNIX, see “Where to go from here” on page 30.

While some of the material covered in this document is applicable to UNIX in general, this document is written with Sun UltraSPARCs in mind.

UNIX Clusters at Notre Dame

Sun UltraSPARCs are multi-user and multi-tasking workstations that run the UNIX operating system. Multi-user means more than one user can use a computer at the same time. Multi-tasking means one user can run more than one program at the same time. With its flexibility and extensive network communication capabilities, UNIX has become a standard operating system for workstations as well as super computers.

Notre Dame Computer Support

For general assistance with computers at the University of Notre Dame, contact the Information Resource Center (IRC), Room 111 Computing Center/Math Building (631-8111). If you know how to use electronic mail (there is a separate course for this), you can send problem reports or questions to the computer account `suggest`. Your messages will be received by a UNIX consultant or system administrator, who manage the computers in the UNIX clusters. In addition, the clusters have student consultants who may be able to help you. Finally, the OIT offers documents to assist you. These documents and an order form are available in the IRC and the campus clusters. To see a list of documents and the text of a number of documents online, or to order documents from our Web site, visit the Documentation homepage at the following URL:

http://www.nd.edu/~doc/

You can take the UNIX II course, as well as other non-credit courses on a wide variety of computing topics, through the OIT. Contact the IRC for more information.

Your Computer Account

As a student of the University of Notre Dame, you are assigned a computer account on the UNIX clusters that enables you to use the Sun UltraSPARCs. With this account, you are issued an afs_id and password. You need to know your afs_id and password to log in. If you do not have this information, contact the IRC or a cluster consultant.

Your password is the main component of UNIX security. It is very important that you do not share it with other users or write it down where somebody else can read it. If you need to share some data in your computer account with other users, the system provides means to allow or disallow access to specific users and to create groups.

Working on the console

A console is the full display area on your terminal screen (without a windowing environment). The term *console* is used here to differentiate the display area from a windowing environment, which usually comprises parts of the screen display area. While learning basic UNIX commands you will be issuing commands from the console. Instructions on how to start a windowing environment will be given later.

Logging In

Before you can use a computer, you need to log in. After you are done using a computer, you need to log out. This prevents other users from gaining access to your account. On a Sun UltraSPARC you will see this prompt on the screen:

```
hostname%login:
```

where hostname is the name of the computer. Each computer is given a name to distinguish it from others. You should also see the name of the computer on a label affixed somewhere on the computer itself.

On rare occasions you may not see the login prompt on the screen. If that happens, press the **Return** key several times. (Ask a cluster consultant for help if you still do not see the login prompt.)

On the terminal screen you will see the cursor, a small gray rectangle that appears after the login prompt. Whatever you type on the keyboard appears at the location of the cursor, and the cursor moves one space to the right.

To log in:

1. At the `login:` prompt, type your afs_id and press **Return**. Pressing **Return** tells the computer that you have finished typing a command or a line of data. All UNIX commands are executed by pressing **Return**. The system displays the `password:` prompt.

2. Type your password and press the **Return** key. For security reasons, your password will not display as you type.

AFS_id and password entry

UNIX is case sensitive so it's important to enter your afs_id and password exactly as shown on your ID application form, including upper and lower case letters.

If the afs_id and password combination you type is incorrect, you will see the following:

```
Login incorrect
```

Repeat the login procedure. If you still cannot log in, ask a consultant for help.

If the afs_id and password are correct, the system will display the message of the day, then the following:

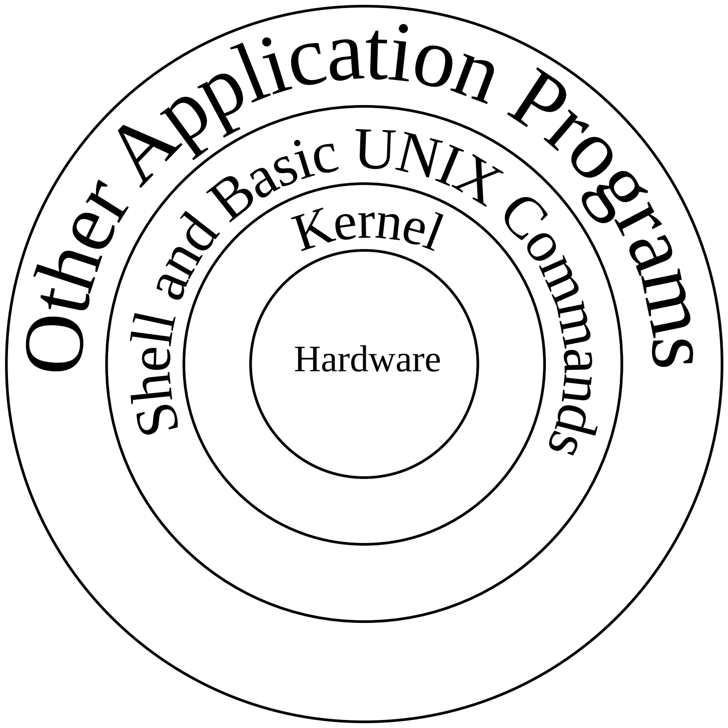
```
OpenWindows? (Control-C to interrupt)
```

3. Press **Control-C** (the **Control** key and the **C** key simultaneously) to interrupt the starting of the OpenWindows environment.

Because this section deals with UNIX commands that you can issue from the console, you want to interrupt the starting of the OpenWindows windowing environment. You are now ready to work on the console.

About the UNIX operating system

The UNIX operating system consists of two primary parts: the kernel and the shell. The kernel is a central program that manages the computer system. The shell is a program that processes your commands and sends them to the kernel.



The kernel controls your computer's hardware devices, such as disks, memory, mouse, monitor, and so forth. In a multi-user environment the kernel makes sure that each user has a fair share of the computer's resources. Most users never work directly with the kernel, but rather work through the shell.

The shell surrounds the kernel in the same way that the shell of a nut surrounds the kernel inside it. You give commands to the shell, not the kernel. The shell processes your instructions and relays them to the kernel. This is why the shell is often called a command interpreter.

The shell/kernel structure allows UNIX users to modify the shell without changing the kernel. In other words, you can use different shells to work with the same kernel. Different shells have their own advantages and disadvantages. For the Sun UltraSPARCs in the OIT clusters the default shell is called the C Shell (csh). When you log in to a Sun UltraSPARC, the C Shell is started for you. When the shell is ready to accept your commands, it displays a shell prompt. The default prompt on the OIT systems is:

```
hostname%
```

Changing Your Password

When you log in for the first time, you should change your password. Make sure you remember it. If you forget your password, you can ask the IRC or your system administrators to give you a new one, but you cannot access your account in the meantime; getting the new password can take up to two days. To change your password, type the following:

```
hostname% passwd
```

NOTE: The command is **`passwd`**, not **`password`**.

You will be asked to enter your old and new passwords. There are some simple rules about choosing a password:

- You should be able to remember it, but it should not be easy for other people to guess.
- It should be six to eight characters.
- It can contain any character: upper-case letters, lower-case letters, numbers, spaces, tabs, and special characters.

Logging Out

When you are done working, you need to log out to prevent other people from using your account. If you do not log out, malicious users could send mail messages using your name or even delete all the data saved in your computer account. To log out, type the following:

```
hostname% logout
```

When you are correctly logged out, the login prompt will return to the screen. Make sure that you see the login prompt before you leave the computer.

Manual Pages

Descriptions of UNIX commands are available online in the manual pages. If you need information on a command, check the manual pages using the command **`man`**. For example, to find information on the command **`passwd`**, type the following:

```
hostname% man passwd
```

Note that here **`man`** is the command and **`passwd`** is an argument to that command. Some commands do not have an argument, others have one argument, and still others can have more than one argument. Some arguments are optional; if you type a command without its optional argument, it uses a default value for the missing argument. In all cases, type a space between a command and its argument and between two arguments. It does not matter if you type more than one space to separate a command and its arguments, but there must be at least one.

After you type **`man passwd`**, the first screen of information about passwd will be displayed. To scroll down one screen, press the space bar; to scroll up the screen again, press **b** (short for back). Note that you do not have to press **Return** here. Pressing **h** (short for help) lists all the available commands to move around in the manual pages. To quit reading about the command **`passwd`**, type **q** (short for quit) and you will be returned to the shell prompt.

Note that the command **`man intro`** produces a list of available UNIX commands. However, this list may not be useful until you are more familiar with the UNIX operating system.

One useful option of the command **`man`** is the **`-k`** option. This option asks man to search for keywords in the descriptions of available commands. It then lists the commands with their synopses. For example, the command

```
hostname% man -k print
```

lists the available commands related to printing.

Files and Directories

A file is a collection of related pieces of information. It can be human-readable (text file) or readable only by the computer (binary file). A directory is a collection of files. It is a good practice to put files of the same type or for the same application in one directory, for example, all documents in a directory called *Doc* and all FORTRAN programs in a directory called *Fortran*. It is also a good practice to name directories with names starting with upper-case letters. A directory can also contain other directories.

When you log in, you will be in a directory called your home directory. In general, you can create files or directories in your home directory, but not in other users’ home directories. Stated another way, you have write permission in your home directory, but not in other users' home directories.

To find out which directory you are in at any time, type **pwd** (short for present working directory). For example:

```
hostname% pwd
/afs/nd.edu/user#/afsid
```

In this example, you are in the directory named */afs/nd.edu/user#/afsid*. The symbol / is used to show different levels of directories. The present working directory resides in the directory */afs/nd.edu/user#*, which resides in the directory */afs/nd.edu*, which resides in the directory */afs*. A directory in another directory is called a subdirectory; the directory where a subdirectory resides is called its parent directory. For example, */afs/nd.edu* is a subdirectory of */afs* and */afs* is the parent directory of */afs/nd.edu*. The directory /, being the parent of all directories, is called the *root* directory. The structure of the directories and files in a computer system is like an inverted tree, with the root at the top and the branches below it (representing other directories and files).

To find out what files or directories are contained in your home directory, use the command **ls** (short for list).

```
hostname% ls
Private      Public      YESTRDAY      www
```

This example shows four entries in your home directory. An easy way to find out more about those entries is by using the **-F** option of the **ls** command.

```
hostname% ls -F
Private/      Public/      YESTRDAY/      www/
```

The / symbol at the end of each of entry indicates that the entry is a directory. If there is no symbol after an entry, it is a file. If there is an asterisk

(*) after an entry, it is an executable file; this means that if you type the name of the file, it will issue instructions for the computer to run.

Since the directories Private, Public, YESTRDAY, and www are in your home directory, their full names include the name of your home directory. These full names are /afs/nd.edu/user#/afsid/Private, /afs/nd.edu/user#/afsid/Public, /afs/nd.edu/user#/afsid/YESTRDAY, and /afs/nd.edu/user#/afsid/www, respectively.

When you are in a directory, you can refer to its subdirectories by their short names. Since you are now in your home directory (/afs/nd.edu/user#/afsid), you can refer to these directories simply as Private, Public, YESTRDAY, and www. For example, you can list the content of /afs/nd.edu/user#/afsid/Public directory by the command:

```
hostname% ls -F Public
```

If this command does not return anything, the directory Public is empty. If you are not in your home directory, you have to refer to the above directories by their complete names.

```
hostname% ls -F /afs/nd.edu/user#/afsid/Public
```

As the names indicate, the directory Private is where you should save your private data. The directory Public is appropriate for data that you allow other users to read. These directories have been set up so that other users can read the files in the directory Public, but they cannot read the files in the directory Private.

The directory YESTRDAY contains the "snapshot" of your home directory at 2:00 a.m. (see “Recovering Accidently Deleted Files with AFS” on page 17). The directory www is used if you access the World Wide Web (WWW). WWW is not discussed in this document. World Wide Web courses are available from the OIT.

Do not tamper with Public, YESTERDAY, or www

Do not remove or rename your Public directory, because it contains some hidden configuration files. In case you accidently delete the Public directory, type
`/usr/local/bin/secure_home_directory`
to fix the problem.

Also, do not remove your www directory. You cannot remove the YESTRDAY directory and it does not take any space from your disk quota.

If your account was created prior to January 1993, you may not have the directories Private and Public. In this case, if you need help working through the rest of this document, contact a consultant or the IRC.

You can get a more extensive listing of your directory by using the **-l** option of **ls**.

```
hostname% ls -l
total 4
drwxr-xr-x        2 afsid     2048 Jun  4 16:09 Private
drwxr-xr-x        2 afsid     2048 Jun  4 16:09 Public
drwxr-xr-x        2 afsid     2048 Oct  4 10:17 YESTRDAY
drwxr-xr-x        2 afsid     2048 Oct  4 10:17 www
```

Briefly, the first line tells you the listed items use 4 blocks of data in the computer file system. The size of a block of data is 1024 bytes (a byte is the size for storage of one character). The other lines tell you, among other things, that the listed items belong to the user, each item is stored in 2048 bytes, and two were created on June 4 of the current year at 16:09 and two were created on October 4 of the current year at 10:17. If the file creation time is longer than six months ago, it is shown in the format "month date year."

You can combine the **-F** and **-l** options of **ls** into either **-Fl** or **-lF**.

```
hostname% ls -Fl
total 4
drwxr-xr-x        2 afsid     2048 Jun  4 16:09 Private/
drwxr-xr-x        2 afsid     2048 Jun  4 16:09 Public/
drwxr-xr-x        2 afsid     2048 Oct  4 10:17 YESTRDAY/
drwxr-xr-x        2 afsid     2048 Oct  4 10:17 www/
```

You can use an asterisk (*) to represent any string of characters. Such a substitution symbol (*) is called a wild card.

```
hostname% ls -F *
Private:
Public:
YESTERDAY:
www:
```

```
hostname% ls -F P*  (upper-case P)
Private:
Public:
```

```
hostname% ls -F p*  (lower-case p)
No match
```

Remember that UNIX is case sensitive (an upper-case F is not the same as a lower-case f).

Changing the Working Directory

To move to another directory, use the command **cd** (short for change directory). For example, to move to the directory Public, type:

```
hostname% cd Public
```

If you are not presently in your home directory, change to the directory Public by typing its complete name as an argument to the command **cd**.

```
hostname% cd /afs/nd.edu/user#/afsid/Public
```

Check where you are now by using the command **pwd**:

```
hostname% pwd
/afs/nd.edu/user#/afsid/Public
```

You can go back to the parent directory of the present working directory by typing two periods:

```
hostname% cd ..
```

The symbol **..** represents the parent directory. Other useful symbols are **.** and **~**. The symbol **.** represents the current directory.

```
hostname% ls -F ./P*
Private:
Public:
```

The symbol **~** represents your home directory. The symbol **~** followed by an afs_id represents the home directory of the person with that afs_id. For example:

```
hostname% cd ~guest1
```

sends you to the home directory of the person whose afs_id is *guest1*. The following command:

```
hostname% ls -F
Private/      Public/      YESTRDAY/      www/
```

lists the contents of your home directory. When issued without any argument, the command **cd** sends you back to your home directory (equivalent to **cd ~**).

```
hostname% cd
hostname% pwd
/afs/nd.edu/user#/afsid
```

Getting Information from a File

Reading a File (cat and more)

You can read a file using the command **`cat`** (short for concatenate). To read the message of the day type the following:

```
hostname% cat /etc/motd
```

If the file is longer than one screen, it will scroll over the screen quickly and you will be able to read only the last screen. In this case, use the command **`more`**, which displays a file one screen at a time. If the whole file can be displayed on one screen, the command **`more`** displays the file and quits, just like the command **`cat`**. If the file is longer than one screen, **`more`** displays only the first screen of the file and waits for further commands. To scroll down another screen, press the space bar; to scroll up the screen, press **b** (short for back). Press **h** (short for help) to list all the available commands to move around in the command more. To quit reading about the command more, press **q.** Try using **`more`** to read the message of the day again:

```
hostname% more /etc/motd
```

Word Count (wc)

You can find out how many lines, words, and characters in a file by using the command **`wc`** (short for word count).

```
hostname% wc /etc/motd
31     237    1997 /etc/motd
```

The response tells you that there are 31 lines, 237 words, and 1997 characters in the file.

Searching for a String (grep)

You can list the lines in a file that contain a specific string by using the command **`grep`** (short for get a regular expression). For example, to list the lines with the string `UNIX` in the file `/etc/motd`, type the following:

```
hostname% grep UNIX /etc/motd
public cluster area for help with "walk-in" UNIX questions
from all
```

Managing Files

Copy (cp)

You can create a copy of a file using the command **`cp`** (short for copy).

```
hostname% cp /etc/motd hello.f
hostname% cp hello.f hello.f.bak
hostname% ls -F
hello.f      hello.f.bak
```

Be careful when you copy files

Copying a file over an already existing file with the same name will overwrite the existing file. You can avoid this potential disaster by using the option **`-i`** (short for interactive) with the command **`cp`**.

```
hostname% cp -i hello.f hello.f.bak
overwrite hello.f.bak?
```

If you answer yes by entering **`y`**, the file hello.f.bak will be replaced with a copy of hello.f. Any other answer will leave the file hello.f.bak untouched and the command **`cp`** not executed.

Move (mv)

You can change the name of a file using the command **`mv`** (short for move).

```
hostname% mv hello.f.bak hello.f.backup
hostname% ls -F
hello.f         hello.f.backup
```

To avoid overwriting an existing file, you can again use the option **`-i`**.

```
hostname% mv -i hello.f hello.f.backup
remove hello.f.backup?
```

As with **`cp -i`**, only the answer **`y`** (lowercase) will replace the existing file.

Delete (rm)

To delete a file, use the command **rm** (short for remove) followed by the name of the file. This command permanently removes the file; you cannot change your mind and get the file back. To delete the file hello.f.backup, type:

```
hostname% rm hello.f.backup
hostname% ls -F
hello.f
```

You can also use the option **-i** here as well:

```
hostname% rm -i hello.f
rm: remove hello.f? n
```

Do not delete the file hello.f!

If you answer with **y** here, you will have to re-create the file hello.f to continue with the exercises in this document!

Using the C shell (csh), you can alias **cp -i**, **mv -i**, and **rm -i** into other commands; for example, you can make **rm** act as if you entered **rm -i** every time you use it. Since we will not discuss how to alias a command here, you should use the **-i** option of **cp**, **mv**, and **rm** each time you use one of these commands.

Exercise: Working with files

The commands **cp**, **mv**, and **rm** also work over different directories. Type the following commands slowly, trying to understand what they do. If there is something that you do not understand, ask a cluster consultant.

```
hostname% pwd
/afs/nd.edu/user#/afsid/Public
hostname% cp -i hello.f ~/Private
hostname% ls -F ~/Private
hello.f
hostname% mv -i ~/Private/hello.f ../dummy
hostname% ls -F ~/Private

hostname% ls -F ..
Private/     Public/     YESTRDAY/     dummy     www/
hostname% rm -i ../dummy
rm: remove ../dummy? y
hostname% ls -F ..
Private/     Public/     YESTRDAY/     www/
```

File Redirection

When you type the command **ls**, you see the output displayed on your console. You can send this output, and the output of other commands, to a file instead of to the console. This process is called file redirection. To do so, you need to use the **>** operator.

```
hostname% ls -F .. > filelist
hostname% cat filelist
Private/
Public/
YESTRDAY/
www/
```

By default, on the Sun UltraSPARCs the redirection will refuse to overwrite an existing file. However, if you work on another UNIX system, make sure that the same condition is in effect; otherwise you will lose the existing file because the new one will overwrite it. To replace the existing file on a cluster workstation, use the **>!** operator.

```
hostname% ls -F .. > filelist
filelist: File exists.
hostname% ls -F .. >! filelist
```

If a file already exists, you can append the output of a command to the end of the file by using the operator **>>**.

```
hostname% cat filelist
Private/
Public/
YESTRDAY/
www/
hostname% ls -F >> filelist
hostname% cat filelist
Private/
Public/
filelist
hello.f
YESTRDAY/
www/
```

Creating and Removing Directories

You create a directory with the command **mkdir** (short for make directory) and remove it with the command **rmdir** (short for remove directory). In order to remove a directory, it must be empty.

```
hostname% mkdir newdir
hostname% ls -F
filelist     hello.f        newdir/
hostname% cp hello.f newdir
hostname% ls -F newdir
hello.f
hostname% rmdir newdir
rmdir: directory "newdir": Object is remote
hostname% rm -i newdir/*
rm: remove newdir/hello.f? y
hostname% rmdir newdir
hostname% ls -F
filelist     hello.f
```

Access Control Lists

The OIT Sun UltraSPARCs store files in a file system called AFS. This system is transparent; that is, regardless of which computer you use to log in, you see the same files in your home directory. AFS uses access control lists (ACL) to determine who can access the information in AFS file space. Access control lists exist for each directory in the system. The owner of the directory can determine who appears on the ACL and what privileges they have.

Do not use chmod

You maybe familiar with the command **chmod**. Be aware that you cannot use this command to change the permissions of your home directory. You must set the access control list for your directories using the command **fs setacl** as explained in this section.

To see the ACL of a directory, use the command **fs listacl** ***directory***.

```
hostname% fs listacl  ~
Access list for /afs/nd.edu/user#/afsid is Normal rights:
  system:administrators rlidwka
  system:anyuser l
  afsid rlidwka
```

```
hostname% fs listacl ~/Public
Access list for /afs/nd.edu/user#/afsid/Public is Normal
rights:
  system:administrators rlidwka
  system:anyuser rl
  afsid rlidwka
hostname% fs listacl ~/Private
Access list for /afs/nd.edu/user#/afsid/Private is Normal
rights:
  system:administrators rlidwka
  afsid rlidwka
```

The access control rights are described below:

Access Control Right	Description
r	permission to read contents of files in the directory
l	permission to lookup (list) contents of the directory
i	permission to insert files into the directory
d	permission to delete files from the directory
w	permission to write files in the directory
k	enables programs to place advisory locks on a file
a	permission to change the ACL of the directory

When you create a directory, the ACL of the new directory is the same as the one associated with its parent directory. To change the ACL of a directory, use the command **`fs setacl`** directory.

```
hostname% mkdir ~/Public/newdir
hostname% fs listacl ~/Public/newdir
Access list for /afs/nd.edu/user#/afsid/Public/newdir is
Normal rights:
system:administrators rlidwka
system:anyuser rl
afsid rlidwka
hostname% fs setacl ~/Public/newdir system:anyuser none
hostname% fs setacl ~/Public/newdir johanes rl
hostname% fs listacl ~/Public/newdir
Access list for /user#/afsid/Public/newdir is Normal rights:
system:administrators rlidwka
johanes rl
afsid rlidwka
hostname% rmdir  /Public/newdir
```

Recovering Accidently Deleted Files with AFS

You may be able to recover accidentally deleted files. AFS backup of your home directory is started at 2:00 a.m. daily. A file deleted after 2:00 a.m. can be recovered by its owner any time before 2:00 a.m. the next day (when the next backup takes place).

The directory YESTRDAY in your home directory is a "snapshot" your home directory as it was at 2:00 a.m. You can copy whatever files or directories you need from YESTERDAY to your home directory. You cannot remove the YESTRDAY directory and it does not take any space from your disk quota.

If you need to recover files which are already deleted from your home directory at 2:00 a.m., send an e-mail message to **suggest**.

Environment Variables

Your shell, in this case *csh*, operates based on variables called environmental variables. To list all your environmental variables, type:

```
hostname% printenv
TERM=sun HOME=/afs/nd.edu/user#/afsid SHELL=/bin/csh
USER=afsid LOGNAME=afsid PATH=/opt/SUNWspro/bin:/usr/ccs/
bin:/bin:/etc:/usr/bin:/usr/etc:/usr/loc al/src/tex.new/
bin:/usr/local/bin:/usr/ucb:/usr/afsws/bin:/usr/bin/X11:/
usr/local/src/gnu/bin:/usr/sbin:/usr/openwin/bin:/usr/local/
src/Island4. 2/bin:/opt/SUNWwabi/bin:/opt/dt/bin:/opt/
SUNWmfwm:/opt/SUNWguide/bin:/us r/X11R6/bin:/usr/local/src/
framemaker5.1/bin:/afs/nd.edu/users/guest1/bi n:.
...
```

To list just one of these variables, type **printenv**, a space, then the name of the variable. For example, to list your most important variable, type:

```
hostname% printenv PATH
/opt/SUNWspro/bin:/usr/ccs/bin:/bin:/etc:/usr/bin:/usr/etc:/
usr/local/sr c/tex.new/bin:/usr/local/bin:/usr/ucb:/usr/
afsws/bin:/usr/bin/X11:/usr/l ocal/src/gnu/bin:/usr/sbin:/
usr/openwin/bin:/usr/local/src/Island4.2/bin :/opt/SUNWwabi/
bin:/opt/dt/bin:/opt/SUNWmfwm:/opt/SUNWguide/bin:/usr/X11
R6/bin:/usr/local/src/framemaker5.1/bin:/afs/nd.edu/users/
guest1/bin:.
```

The variable **`PATH`** determines the order in which the computer looks for the executable files to execute your command. For example, when you type:

```
hosthame% ls
```

the computer will try to find:

```
/opt/SUNWspro/bin/ls
```

When it cannot find it, it looks for:

```
/usr/ccs/bin/ls
```

After it cannot find it again, it finds:

```
/bin/ls
```

and executes it.

Notice the dot `.` at the end of your PATH environment. This means that the current directory is the last place the computer looks when it is trying to find an executable file.

This issue becomes important when there is more than one executable file with the same name on the computer. In this case the computer executes the one it finds first; it never gets to the rest of the executable files with the same name. You may expect to run a specific executable file, but in fact run a different one due to the setup of your PATH variable. To find out which executable file is run when you issue a command, use the command **`which`**.

```
hostname% which ls
/bin/ls
```

WARNING

Do not run a program that does not reside in one of the directories listed in your **`PATH`** environment unless you know exactly what it does. Even though the name of the program is harmless, it may produce unexpected, even harmful results.

Printing

The command **`lp`** (short for line printer) prints your files. Like the computers, the printers also have names, indicated by the labels attached to them. The available printers are shown in the table below:

Destination	Location
eng_lab1 to eng_lab7	laser printers #1-7 in the Fitzpatrick Hall of Engineering cluster (these printers are labeled in the clusters)
eng_color1	color laser printer in the Fitzpatrick Hall of Engineering cluster
nieuw_lab1	laser printer in Nieuwland Science Hall, room 203
nieuw_lab2	laser printer in Nieuwland Science Hall, room 203
math_lab	laser printer #2 in room 210 of the Computing Center/Math Building

Each computer uses one of the printers as its default printer. If you move to another computer, the default printer may change.

To print the file *hello.f* to the default printer, type the following:

```
hostname% lp hello.f
```

If the default printer is busy, you can print to another printer by using its name (you will usually just print to the default printer). To print to **englab4**, for example, type:

```
hostname% lp -d eng_lab4 hello.f
```

WARNING: Do not print a binary file

Do not use the **`lp`** command to print a binary file. Printing even a small binary file can cause the printer to waste hundreds of pages of paper and may jam the printer.

Since printers are shared by many users, you sometimes have to wait in a queue. To check your print job, use the command **`lpstat`**.

```
hostname% lpstat
eng_lab4-008    breathed.helios.nd.edu! afsid    1342
Jun 11 22:33 on eng_lab4
```

The output of the command **`lpstat`** above shows that your print job has the job number englab4-008. You can cancel that print job by issuing the following command:

```
hostname% cancel eng_lab4-008
```

To cancel all of your print jobs, type:

```
hostname% cancel -u afs_id
```

where ***afs_id*** is your afs_id.

You can change your default printer by setting the environmental variable **LPDEST**:

```
hostname% printenv LPDEST
eng_lab6
hostname% setenv LPDEST eng_lab4
```

Now your default printer is **englab4**, and you do not have to use the argument **-d eng_lab4** with your lp command as long as you stay logged in. However, you now have to use the argument **-d eng_lab6** to print to the other printer.

Compiling and Running a Simple FORTRAN Program

In this section you will copy a file with two lines of FORTRAN 77 statements to your current directory. The way the program is written now, the statements will not be recognized by the computer. You need to translate the statements into something that the computer understands. This translation process is called compilation and the tool used to perform the process called a compiler. To compile the FORTRAN 77 program, you use the FORTRAN 77 compiler (f77).

In this section you will learn how to compile a program and run it. Note that the choice of FORTRAN as the programming language is arbitrary; most programs written in other languages are treated (compiled) in a similar way.

1. Copy the file */usr/local/courses/OIT/unixI/hello.f* to your current directory.

2. To compile the program *hello.f*, type:

   ```
   hostname% f77 -o hello hello.f
   hello.f:
    MAIN:
   ```

   This command compiles the program *hello.f* and saves the executable file under the name *hello* (**-o** is short for output).

3. List the contents of your current directory:

```
hostname% ls -F
filelist     helios     hello*     hello.f:
```

   NOTE: Remember, an asterisk indicates an executable file.

4. You can now run hello.

```
hostname% ./hello
Hello World, FORTRAN style
```

Background Processes

In UNIX terminology a process is simply a program that is running. When you log in to your Sun UltraSPARC, the system starts the C shell. Hence, your first process is csh. The shell then waits for you to issue a command, which the shell runs for you as another process.

Some processes wait for you to enter data from the keyboard and require you to read the output displayed on the screen. In the meantime, you watch and wait for the process to finish. Such a process is called a *foreground* process.

Some processes take a long time to finish, do not require input from the keyboard, and do not display any output that you need to read (for example, a program that performs extensive calculations and may run for hours). Such processes can be run in the *background*. While processes run in the background, you can issue other commands to the shell. To run a command in the background, put an ampersand (&) at the end of the command line. For example, to list the users who are currently on local computers, use the command **`rusers`**. To run it in the background, type the following:

```
hostname% rusers &
[1] 16420
```

The numbers displayed after you issue a background process are the logical process number and the process ID. The logical process number is the number of the background process that you have running; for the first background process it is [1], for the second it is [2], and so on. The process ID is the number the system uses to identify the process. For the above example the logical process number is 1 and the process ID is 16420. When a background process ends, the computer gives you a message telling you either that the process is done or that it terminated because of a problem.

You can list your processes (foreground and background) with the command **ps** (short for process status).

```
   hostname% ps
   PID  TTY      TIME CMD
   5478 pts/5    0:00 rusers
   5356 pts/5    0:01 csh
```

The command **ps** lists your processes together with their process IDs (PIDs). If you use the **-x** option of **ps**, you will see that csh is also listed. You can use a process ID to stop a process by using the **kill** command with the process ID as its argument. For example:

```
hostname% kill 5478
[1]    Terminated               rusers
```

kills process number 16420, which is **rusers**. You can kill only the processes that you own.

In csh, there is another way to list and kill background process:

```
hostname% rusers & jobs
[1]   Running              rusers
hostname% kill %1
[1]    Terminated           rusers
```

If you start a process as a foreground process and want to move it to the background, you can suspend it first with **Ctrl-Z** and then use the command **bg** to move it to the background.

```
hostname% rusers
^Z
Stopped

hostname% jobs
[1]   Stopped              rusers
hostname% bg %1
[1]    rusers &
```

To move a background process to the foreground, use the command **fg**:

```
hostname% fg  %1
rusers
```

OpenWindows

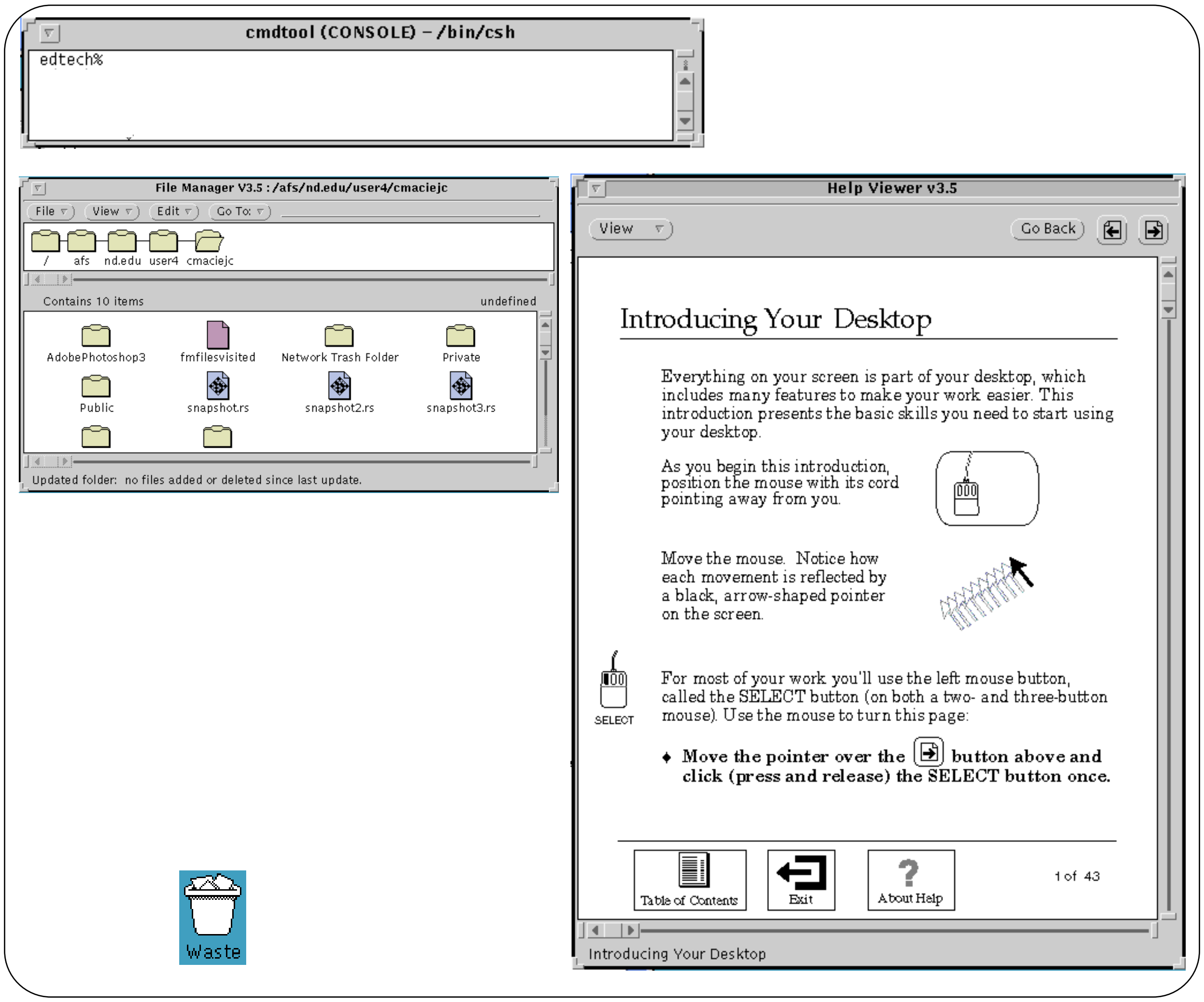
When you work on the console, you work on the whole display area of your terminal screen. In a windowing environment, you can work in a window that only uses part of the screen--and you can open more than one window. The default windowing environment on the Sun UltraSPARCs is OpenWindows.

About the OpenWindows environment

To start OpenWindows, type the following:

```
hostname% openwin
```

After OpenWindows finishes its initialization, you will see a screen with a blue background and the three windows as shown in the figure below.



```
edtech%
```

Updated folder: no files added or deleted since last update.

Introducing Your Desktop

Everything on your screen is part of your desktop, which includes many features to make your work easier. This introduction presents the basic skills you need to start using your desktop.

As you begin this introduction, position the mouse with its cord pointing away from you.

Move the mouse. Notice how each movement is reflected by a black, arrow-shaped pointer on the screen.

For most of your work you'll use the left mouse button, called the SELECT button (on both a two- and three-button mouse). Use the mouse to turn this page:

♦ **Move the pointer over the button above and click (press and release) the SELECT button once.**

Introducing Your Desktop

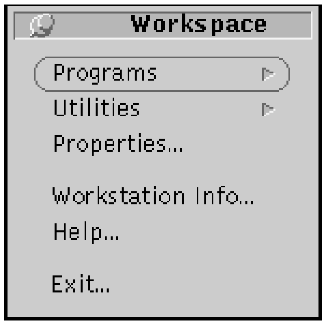
The workspace contains three windows (cmdtool, Help Viewer, File Manager), the Waste icon, and a pointer. The pointer is a symbol whose movement is controlled by the movement of the mouse. With the pointer you operate the OpenWindows menu system and work with your windows. The icon represents a window that is closed. If you move the pointer over the icon and click with the **Left** mouse button, the window opens to display its contents.

The windows you see are described below:

- **cmdtool (CONSOLE)**. The cmdtool window is your Console in OpenWindows. You can type UNIX commands and also see system messages. For example, if you are having network problems, an error message will appear in the Console window.
- **File Manager**. This window displays your directory structure using folder icons to represent directories and icons that look like a page to represent files.
- **Help Viewer**. This window accesses the OpenWindows tutorials. The default tutorial that appears when you first start OpenWindows is *Introducing Your Desktop*, which explains how to use the mouse pointer and mouse buttons.

Workspace menu

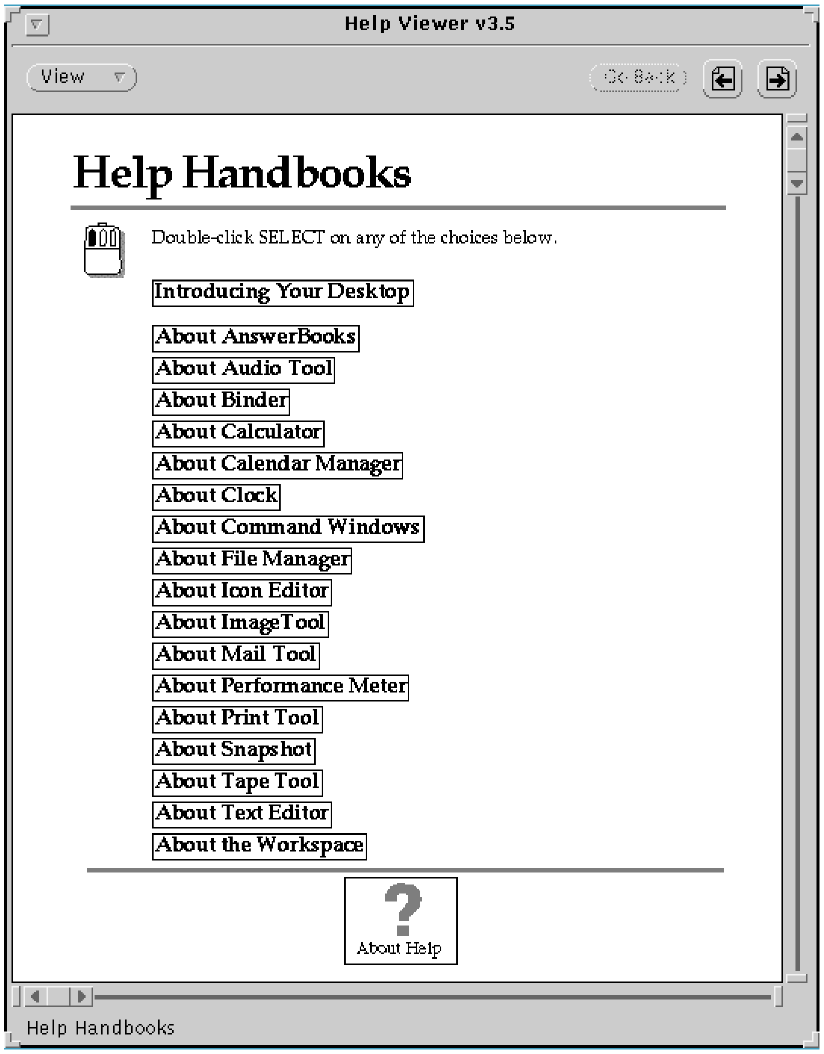
The OpenWindows workspace is the area behind the windows. You can access the main Workspace menu by pressing the **Right** mouse button while the pointer is in the workspace.



To choose an item from the Workspace menu, keep the **Right** mouse button pressed down and move the mouse pointer up or down within the menu. When the desired item is highlighted, release the mouse button. Items with > to the right of the item will display a submenu when selected. Items with an ellipsis (...) to the right of the item will open a window when selected.

OpenWindows Tutorials

For indepth information on how to use OpenWindows, use the tutorials in the Help Viewer window. To see a list of tutorials, double-click on the Table of Contents button at the bottom left corner of the window. A button titled More Handbooks replaces the Table of Contents button. Double-click on More Handbooks to see a list of tutorials.



Help Handbooks

Double-click SELECT on any of the choices below.

- Introducing Your Desktop
- About AnswerBooks
- About Audio Tool
- About Binder
- About Calculator
- About Calendar Manager
- About Clock
- About Command Windows
- About File Manager
- About Icon Editor
- About ImageTool
- About Mail Tool
- About Performance Meter
- About Print Tool
- About Snapshot
- About Tape Tool
- About Text Editor
- About the Workspace

About Help

If the Help Viewer window is not open and is not shown as an icon on your screen, you can start it by choosing **Help** from the Workspace menu.

You can explore these tutorials at your leisure.

Using the Text Editor

The default editor in the OpenWindows environment is Text Editor. You can use the Help Viewer to learn more about Text Editor. Double-click on **About Text Editor** in the Help Viewer window to start the Text Editor tutorial.

In addition to this tutorial facility, the interactive manual pages display utility (answerbook) enables you to find information about specific commands. To start answerbook, type the following in the Console window:

```
hostname% answerbook &
```

Note that the ampersand (&) tells the program *answerbook* to run in the background so that you can still use the Console window for other tasks. The program answerbook is easy to use; just follow the instructions displayed.

Exercise: Practicing what you have learned

In this section you will put your new skills into practice. You will write a program that prints two columns of numbers, where the numbers in the second column are the tangents of the corresponding numbers in the first column; then you will use the graphing program **xvgr** to display them. The objective is to edit a program in one window, compile and run it in another window, and display it in the third window.

1. Use Text Editor to create the program. Type the following lines as shown. Notice the repetition of a pattern in the lines below; use the copy procedure instead of typing the lines over and over again.

```
      do 10 i=1,157
        x=i*0.01
        y=tan(x)
        write(*,*) x,y
 10   continue
      do 20 i=158,471
        x=i*0.01
        y=tan(x)
      write(*,*) x,y
 20   continue
      do 30 i=472,628
        x=i*0.01
        y=tan(x)
        write(*,*) x,y
 30   continue
      end
```

2. Save the program to a file *test.f*. Keep the editor window open, because you will need to edit the program again. Compile the program in another window:

```
hostname% f77 -o test test.f
```

At this stage your compilation may fail if you have typing errors in your program. If this is the case, edit the program. (See the Hint on page page 29 if you need help.)

3. After you have successfully compiled your program, you can try to run it by typing:

```
hostname% test
```

but the command will produce nothing. However, it will run if you type the following:

```
hostname% ./test
```

Two columns of numbers will appear. Do you know why it worked this time? (Hint: Type **`which test`** and check your **`PATH`** environment.)

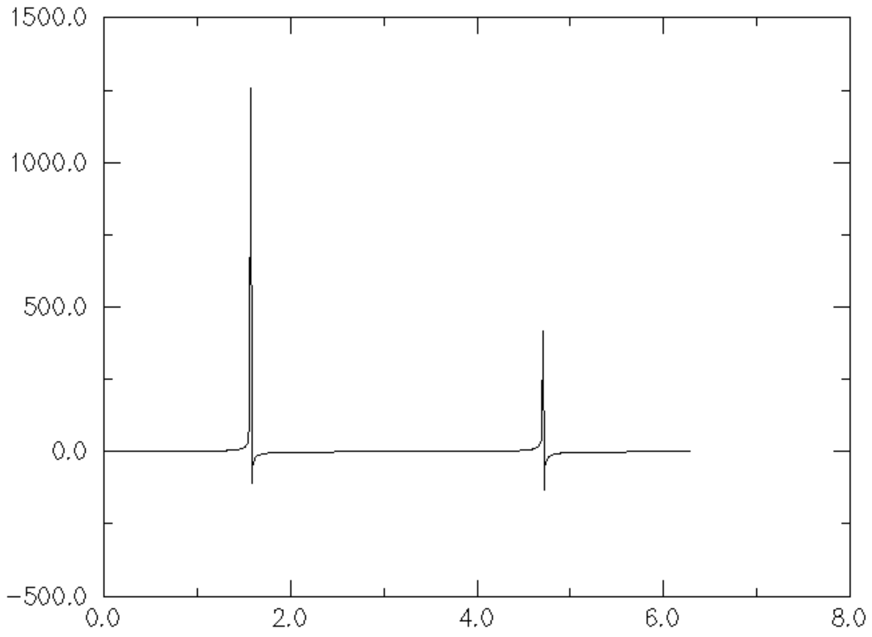
4. Save the output of **`./test`** to a file:

```
hostname% ./test > test.out
```

5. Now plot it with the program **`xvgr`**:

```
hostname% xvgr test.out &
```

You will see a graphic of a line with spikes at two points (as shown in figure below).



The points where tan(*x*) go to infinity are the singular points of that function. If you can assign the values of *x* far enough from the singular points, you will be able to get a graphic in which you can see the values of tan(*x*) at other points. To get a better looking plot, you need to modify the program slightly.

6. Close the xvgr window and edit the file *test.f* by changing these lines:

   ```
   do 10 i=1,157
   do 20 i=158,471
   do 30 i=472,628
   ```

   to

   ```
   do 10 i=1,156
   do 20 i=159,470
   do 30 i=473,628
   ```

7. Repeat the compilation and the running of **`./test`**. You will have a problem creating a new version of *test.out*; either delete the old file or create an output file with a different name (e.g., *test2.out*). Run **`xvgr`** again on the new output of **`./test`**. Now you see a better looking plot. You can try some other things with this little exercise (plotting *x* versus sin(*x*), for example).

8. You can now print your xvgr graphic. From the **File** menu choose **Printer Setup** and print the file with the command **`lp`**. You can also save it to a file.

9. Exit OpenWindows by choosing **Exit** from the Workspace menu.

10. Log out by typing the following:

    ```
    hostname% logout
    ```

    Wait until you see the login prompt again before you leave the computer.

The OIT document titled *U1030 Introduction to the UltraSPARC Clusters* contains more information on UNIX commands, information about available software packages, and other specific information about the Sun UltraSPARCs in the OIT clusters. Request a copy from the IRC or at one of the Sun UltraSPARC clusters.

Hint for program test.f

If you've checked your lines and they appear correct, you may not have spaced over properly. Each line of code must start in the 7th (or higher) column, which means you must hit the space bar at least 7 times. The numbers 10, 20, and 30 on the continue lines, must appear within the first 6 columns.

Where to go from here

After working through this UNIX I document, you should be ready to take the UNIX application courses, although it is highly recommended that you go through this document and practice again before you attend the classes. If you have time, run the tutorials provided in the OpenWindows environment. In addition, as people use dictionaries when they learn foreign languages, you should use the manual pages frequently. Use either the command line version (by typing **`man`** ***`command`*** at the shell prompt) or answerbook.

Resources available through Notre Dame

To complete your introduction to UNIX, consider taking the UNIX II course. Among other things, this course covers using electronic mail (e-mail), reading and writing articles on Usenet News, surfing the net via WWW, and transfering files using the File Transfer Protocal (FTP).

E-mail

People in the computer community communicate with one another using e-mail. You can send a message or even an electronic copy of your technical paper across the world in a short time. Some people use e-mail for fun, like playing chess with opponents they have never seen in person. Some use it for other purposes, such as a professor sending homework assignments to students. On the Sun UltraSPARCs, you can report problems, ask questions, or give suggestions by sending e-mail messages to the suggest account.

UseNet News

Usenet News newsgroups are forums where computer users exchange information on specific topics, ranging from UNIX to investing in the stock market, from C programming to basketball news. For example, on the newsgroup *comp.unix.questions*, people in the computer community post articles asking questions about UNIX and have experts answer them. Some of the solutions given by these experts are unique; they are not documented in manuals or books but gained through years of experience and exploration. Most newsgroups have lists of Frequently Asked Questions (FAQs). You will want to make sure that your question is not in the FAQ; other newsgroup readers do not appreciate reading the same questions repeatedly. There are also daily newspaper articles on the Usenet News under the clarinet group (clari).

World Wide Web

The World Wide Web provides one of the most useful systems for accessing and distributing information on the Internet. Netscape is one of the most commonly used browsers for UNIX users who access Web resources.

The OIT has developed an electronic information service for the University community to share schedules, course and departmental information, and individual home pages on the Web. The Notre Dame Home Page is located at **http://www.nd.edu/** and easily links articles of campus-wide interest. A number of University departments post materials to the Web, including the different colleges, specific academic and administrative departments, and various campus organizations.

The Notre Dame Web site also gives access to information throughout the world by providing links to other universities, government agencies, businesses, and more. On-line help and search tools are available on the Notre Dame Home Page so that users can easily search for relevant material.

FTP

Using FTP, you can transfer files from one computer to another. In the computer community there are public domain programs, software packages, documents, tutorials, and other information available without charge. These files are stored in anonymous FTP sites all over the world. So, once you know how to transfer files using FTP, you will have access to thousands of programs or documents.

Cluster and OIT Documentation

The UltraSPARC clusters have UNIX manuals that you can check out to further your knowledge of UNIX. See a consultant at the clusters for more information. The OIT also offers documents to assist you. These documents and an order form are available in the IRC and the campus clusters. To see a list of documents and the text of a number of documents online, or to order documents from our Web site, visit the Documentation homepage at the following URL:

http://www.nd.edu/~doc/

Outside Resources

In addition to these manuals, there are UNIX books that you can buy. Some of them are better than others. Glance at the introduction and browse through a book before you buy it to make sure that it is at the right level for you and that you like the style. The following are good introductory UNIX books:

1. Mitchell Waite, Donald Martin, and Stephen Prata, *UNIX Primer Plus*, Howard W. Sams and Company, ISBN: 0-672-22028.

2. A. N. Walker, *The UNIX Environment.*, John Wiley Sons, ISBN: 0-471-90564-X.

3. Paul W. Abrahams and Bruce A. Larson, *UNIX for the Impatient*, Addison Wesley, ISBN: 0-201-55703-7.

4. Elizabeth A. Nichols, Sidney C. Balin, and Joseph C. Nichols, *UNIX Survivor Guide*, Holt, Rinehart Winston, ISBN: 0-03-000773-9.

5. Mike Loukides, *UNIX for FORTRAN Programmers*, O'Reilly and Associates, ISBN: 0-937175-51-X.

When you become more experienced, you may want to subscribe to a few magazines, such as *UNIX Review*.