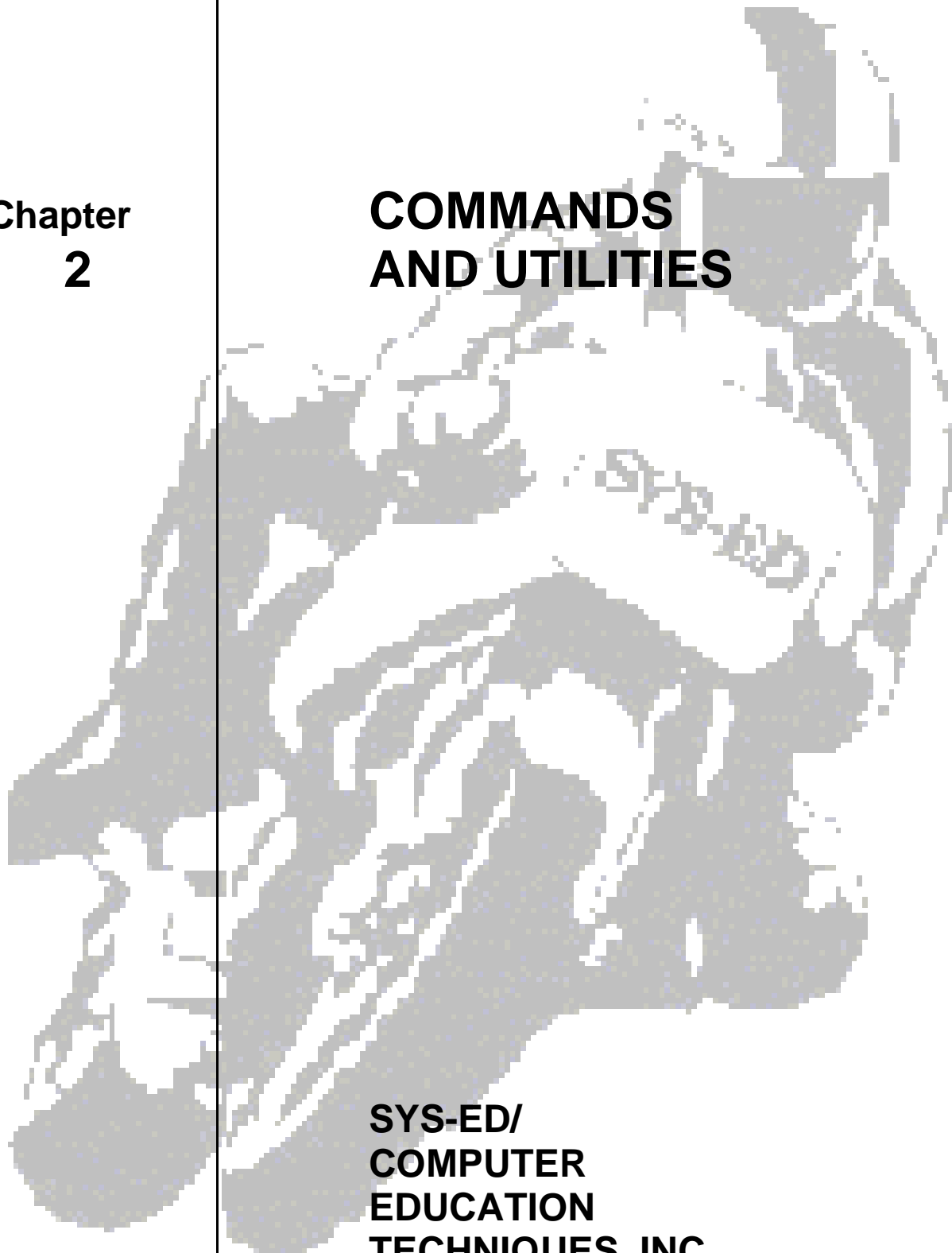


**Chapter  
2**

**COMMANDS  
AND UTILITIES**



**SYS-ED/  
COMPUTER  
EDUCATION  
TECHNIQUES, INC.**

**Objectives**

You will learn:

- C Common UNIX commands.
- C How to perform standard operations on files and directories.
- C How to run common UNIX utilities.
- C How to write shell scripts.
- C Shell arguments and how they are processed by the shell.
- C Shell environment variables and how to manipulate them.
- C How to configure the user environment at login time.
- C How to write shell functions and intercept shell interrupts.

---

## 1 Getting Started

---

### 1.1 Logging in to a Solaris System

Follow the appropriate instructions for accessing the UNIX server.

UNIX will usually display a message such as:

login:

The password will not be displayed on the screen as you enter it.

login: abcdef  
password:

Syntax and characteristics of the UNIX operating system include:

- C UNIX systems are case sensitive; upper and lower case characters are treated as different.
- C All UNIX type systems generally use lowercase characters.
- C All commands should be typed in lowercase characters.
- C All users on a UNIX system are assigned their own usernames.
- C When users are logged on to the system, they are placed in a HOME directory, which is a portion of the disk space reserved just for them.

---

## 2 File/Directory

---

### 2.1 pwd - print working directory

When a user logs in to a UNIX system, they are located in their own directory space. Users are generally located off the /usr directory.

The pwd (print working directory) command displays the pathname of the current directory. This will be helpful for ascertaining the user's current location.

---

### 2.2 ls - list files

The ls command is similar to DIR in DOS; it displays a list of all the files in the directory.

When the shell prompt reappears and there is no file listing of the directory contents, this does not necessarily mean that no files are stored there. UNIX systems support hidden files.

UNIX systems extend the functionality of its commands by using special flags or switches. Switches are preceded with a - symbol.

#### Example:

```
$ ls -la
```

The switch l stands for long listing, and the a switch is for all files, including directories and hidden files.

The first part reveals the permission settings of the files (-rw-r--r--).

The codes listed are:

d	directory	r	read	w	write	x	execute	-	no access
---	-----------	---	------	---	-------	---	---------	---	-----------

There are three sets of permission settings:

- C One set for the owner of the file.
- C One set for the group to which the user belongs.
- C Another set for other users on the system.

```

| specifies whether a directory entry
|
| permissions for the owner of the file
|
| permissions for group members
|
| permissions for others
|
- rw- --- ---

```

---

## 2.3 cat - concatenate

The cat utility is used to combine files or typing files to the screen.

The cat command sends its output to the console screen, which in UNIX is called standard out or stdout.

**Example:**

```
$ cat .profile
```

The format of this command specifies that the cat utility is to use the file .profile as its input, and that the output be sent to the console screen.

The cat command can be used to copy data from the keyboard to a file.

**Example:**

```
$ cat - > temp
```

The direct translation is that the input is from stdin (- means the keyboard), and that the output is written to the file called temp (> means dump into).

**Example:**

Type the following lines of text; ignore any typing mistakes, and do not use the cursor arrow keys.

```
Small change got rained on by his own 38 down by the arcade,  
The marquees weren't weeping, they went stark staring mad,  
and the cabbies were the old ones who really had it made
```

Press CTRL d to terminate text entry, and the shell prompt returns.

The cat utility can also be used to join many files together into a single file.

**Example:**

```
$ cat temp .profile > temp1
```

This command copies the two files (temp and .profile) to a file called temp1.

**Example:**

```
$ cat temp >> temp1  
cp (copy)
```

The >> is used for appending to an existing file.

**Example:**

```
$ cp .profile temp2
```

This command stands for copy, and is used for copying one file to another.

It copies the file .profile to another called temp2. If temp2 had already existed, its previous contents would've been erased.

Files can also be copied to another directory. The command (do not type this) \$ cp \* /usr/tmp would copy all the files in the current directory to the directory /usr/tmp.

---

## 2.4 rm - remove

The rm utility is used for erasing files and directories.

**Example:**

```
$ rm temp2
```

This removes the file. Once a file is removed, it cannot be restored.

To cover situations where mistakes might occur, a switch -i appended to this command will request a Yes or No response before deleting the file.

**Example:**

```
$ rm -i temp1
```

Take note that switches are written before the filenames.

---

## 2.5 head

The head command is used to view the first few lines of a file. It accepts a switch specifying the number of lines to view.

**Example:**

```
$ head -2 temp
```

This command would list the first 2 lines of the file temp on the console.

---

## 2.6 tail

The tail command is used for viewing the last few lines of a file. It accepts a switch specifying the number of lines to view.

**Example:**

```
$ tail -2 temp
```

The command lists the last 2 lines of the file temp on the console.

---

## 2.7 wc - word count

The wc utility displays a count of the number of characters, words and lines in a file.

**Example:**

```
$ wc temp
```

It will print out a list of the number of lines, words, and characters respectively.

The switches for this utility are:

-l	print line count	-c	print character count	-w	print word count
----	------------------	----	-----------------------	----	------------------

---

## 2.8 grep

The grep command searches a file for a pattern.

**Example:**

```
$ grep 'change' temp
```

This command searches the file temp in the current directory for the text string 'change'.

Using the \* wildcard specifies all files, and using the switch -n will display on which line the text string is found.

**Example:**

```
$ grep 'hello' *
```

This command searches all files in the current directory for the text string 'hello'.

---

## 2.9 mv - move

The mv command is used for moving or renaming files.

**Example:**

```
$ mv temp temp2
```

This renames the file temp to temp2.

**Example:**

```
$ mv temp2 /usr/tmp
```

The mv command would move the file temp2 into the directory /usr/tmp. It would no longer appear in your home directory.

### 3 Directory Management

UNIX systems support hierarchical directory structures.

The commands used for directory maintenance are:

pwd	print current working directory.
cd	change directory.
mkdir	make a subdirectory.
rmdir	remove a subdirectory.

#### 3.1 pwd - print working directory

The pwd command prints the current working directory on the console screen.

In UNIX, the hard disk area is divided into directories, much like a book is sub-divided into chapters and paragraphs.

The directories form a hierarchical level, which simplifies the organization of the files on the system.

The top most directory in UNIX is called root, and consists of a number of subdirectories grouped according to function.

When you use the ls command to list the file contents of a directory, those entries which are directories are preceded with a 'd' character.

The following sample screen illustrates this:

```
ls -la
total 19 files
drwxr-sr-x   3   b_brown   512   Nov 24 12:05   .
drwxr-sr-x  46   root       1024  Nov 23 16:46  ..
-rw-r--r--   1   b_brown  2501  Mar  3 1992   .profile
drwxr-s---   2   b_brown   512   Nov 24 12:05   datafiles
-rw-r-----   1   b_brown    0     Nov 24 12:05   dirlist
-rw-r-----   1   b_brown   24    Nov  5 08:59   tmp
```

Notice that datafiles is a sub-directory. There are two other subdirectory entries in the listing.

```
.. the current directory
.  the parent directory (upwards towards /)
```

---

### 3.2 cd - change directory

The cd command is used to change the current directory.

#### Examples:

```
cd .
```

```
cd ..
```

```
cd /
```

```
cd $HOME
```

Try entering the above commands one at a time and then use the correct UNIX command to determine what the current directory is before proceeding to the next one.

---

### 3.3 mkdir - make directory

The mkdir command makes a sub-directory under the current directory.

---

### 3.4 rmdir - remove directory

The rmdir command removes/deletes a sub-directory under the current directory.

---

## 4 Printing

---

### 4.1 lp - line printer

The lp command is used for obtaining a hardcopy printout of a file.

**Example:**

```
$ lp temp
```

---

### 4.2 pr - print

The print utility displays the file to stdout.

The switches it accepts are:

-ln	set page length in lines to n.
-wn	set line length in characters to n.
-n	number output lines.
-t	do not print page header or trailer.
-digit	number of columns to use.
+digit	start at page number.

**Example:**

```
$ pr -2 -n -l20 temp
```

This prints the file temp to stdout, using two columns per line, numbering each line, using a page length of 20 lines per page.

---

## 5 Network

---

### 5.1 who

The who command displays a list of the people registered on the system.

The -u switch displays a list of those people currently logged into the UNIX server.

**Example:**

```
$ who -u
```

---

### 5.2 talk

The talk command sends interactive messages between users.

**Example:**

To identify a user from the list printed by the previous who command, enter the command, but this time replace username with the name of an actual user:

```
$ talk username
```

Exchange a few messages. When a user receives a message, they should respond.

The interactive exchange is canceled by using the Ctrl D keys; some systems may use Ctrl C. When this occurs, the message end of message appears on the other terminal, and control is returned to the shell.

Terminate the message exchange by pressing Ctrl D or Ctrl C on other systems.

---

### 5.3 mesg - message

This command permits or denies messages invoked using write.

The switches it uses are:

-y and -n

Type the following command to disable messages.

```
$ mesg n
```

Type the following command, replacing username with your UNIX login name:

```
$ talk username
```

---

### 5.4 passwd - password

Each user in UNIX is assigned a password. This is changed using the passwd utility.

**Example:**

```
$ passwd
```

The utility asks for the new password twice, to make sure that you didn't misspell it. The user types in the new password, but notice that it will not be displayed on the screen. It is done this way for security purposes.

The new password would then be in effect. If the user were to log in again, the system would ask for this password to be typed before allowing access.

---

## 6 Other Useful Commands

---

### 6.1 cal - calendar

The cal command prints a calendar to the screen.

---

### 6.2 man - manual

The man command locates and displays the reference page for a specified item.

**Example:**

```
$ man df
```

The `:` on the screen represents a prompt. Pressing enter will continue the display.

To quit from the manual, type `q`.

---

### 6.3 set

Typing the set command will print out a list of all the current shell environments. This is a list of all the variables and settings that the shell currently has.

The majority of these are loaded when you first log in. The first thing the shell does is execute the file `.profile` in the user home directory.

**Example:**

```
$ set | more
```

---

## 6.4 Shell Variables

The shell supports user and system variables.

### Examples:

PS1	shell prompt
HOME	home directory

PS2	secondary line shell prompt
LOGNAME	login name

Variables are assigned values using the = sign.

### Examples:

```
$ PS1="Hi there :"
```

This changes the system prompt PS1 from a \$ to the text 'Hi there :'

```
Hi there : dir="ls -la"
```

This variable dir can now be used as a command. Before executing it, the shell will expand it out. Shell variables, when used as commands, are preceded with a \$ symbol.

```
$ dir
```

The shell expands this out to the command 'ls -la' then executes it, resulting in a long listing of the current directory.

---

## 6.5 echo

The echo command echoes what follows.

### Examples:

```
$ echo "hello world"
```

```
$ sample="Hello world"
```

```
$ echo $sample
```

---

## 7 Background Processing

Another feature of UNIX is running processes in the background. When a command is appended with the & symbol, it is run in the background and the shell prompt immediately returns.

This is useful for such things as printing documents, because it allows something something else to be performed instead of waiting for the document to conclude printing.

Each process in UNIX is identified by a PID - process identification number. When a background process is started off, UNIX will print its PID number in square brackets.

### Example:

```
$ ls -l > dirlist&
```

Take note that the shell prompt immediately returns, and a number in brackets is printed.

The & character signals to the shell that the command is to run in the background.

```
2868  
$
```

This is the PID number of the background task. There can be more than one background task. Sometimes, it may be necessary to shut down a background task. The kill command is used to shutdown a task, and it accepts the PID number of the task to shutdown.

---

### 7.1 sleep

The sleep command suspends execution for a specified time interval, expressed in seconds.

### Example:

```
$ sleep 200&
```

This will run the sleep program in the background for 200 seconds.

---

## 7.2 pstree - process status, tree display

The pstree command reports on the active processes being run.

### Example:

```
$ pstree -p
```



---

## 8 Piping Commands

UNIX systems offer pipelining. This means that the output of one utility can be fed directly into the input of another.

The | symbol indicates pipelining in UNIX systems.

### Example:

```
$ pr -2 -n temp | wc -l
```

This creates a pipe between the output of the pr utility and the input of the utility wc.

The command runs pr, which takes the file temp, adds lines numbers to it and formats it using two columns per line, feeding the output into wc which prints the line count on the stdout.

A pipe inter-connects two programs together.

A pipe is used when a command (program) is used on both sides. In this example, pr and wc are both commands, so they are connected using a pipe. If the right side was a filename or standard-device name, then you use a redirection symbol to connect the program to the device-name or file.

---

**9 DOS Commands and UNIX: Equivalence**

DOS Command	UNIX Command
dir	ls
cls	clear
del	rm
copy	cp
move / rename	mv
type	cat
cd	cd
more < file	more file
md	mkdir
rd	rmdir
win	startx

Unlike DOS, UNIX commands and their arguments MUST be separated by a space.