Lecture 19: Input/ Output Redirection & Filters

Lecture Overview

- lnput / Output Redirection in UNIX
- Input redirection Output redirection
- Standard error redirection
- Pipes and Filters
- grep
- sort
- cut
- paste
- Examining file contents
- more
- pg
- tee
- tr

. Objectives

After studying this chapter, the students should be able to:

- Understand Input / Output redirection in UNIX
- Understand the concept of standard input, standard gutput and standard error files
- Understand combining commands through pipelines
- Use commonly used filters for problem solving

Input / Output Redirection in UNIX

Design philosophy of UNIX commands

Single function

No assumptions about source of input, destination of output and error messages

Standard files of UNIX: stdin, stdout, stderr

Shell's ability to assign the I/O devices

In UNIX, when a user requests execution of a command by typing it at the prompt, the shell interprets the command, executes it through the kernel and displays the result. .

By default, the command reads the input from the standard input file, gives the output to standard output file and the errors to standard error file.

The shell in turn assigns default devices to these files; standard input to the keyboard, standard output and standard error to the terminal. So when a command is described as reading from the standard input and writing to the standard output, that means it takes its input from the keyboard and sends output to the terminal.

Input / output redirection is the ability of the shell to change the default assignments, using redirection operators.

.Input Redirection

Changing the default input source

The input redirection operator: u<u

$ command < filename

After the execution, standard input is reassigned to the default devices

Input redirection changes the assignment for standard input. The '<' (less than symbol) operator makes the filename following the it the new standard input.

Example: The following commands will produce the same output:

$ cat emp.dat <Enter>

$ cat < emp.dat <Enter>

The difference being that in the first command, cat will open and read the file emp.dat whereas in the later, cat will *read* from *the* standard *input,* which is redirected to the emp.dat file by the shell. .

$ cat <Enter>

Hello! Welcome to RU <Enter>

Hello! Welcome to RU

This is a test of input and output <Enter>

This is a test of input and output <Enter>

^d
$

In the above command cat without a filename or a redirection will read the input from the standard input device, the keyboard, and displays the contents onto the standard output device, the terminal.

On pressing CTRL + 0 (control and D), the process terminates. The sequence of CTRL and 0 denotes the end of file *fot* ordinary files.

.Output Redirection

Changing the default destination of output
The output redirection operator: n>n
$ command> filename
After the execution, the standard output is reassigned to the default device

The output redirection changes the default destination of the output. It is represented by the n>n (greater than symbol). It is useful in storing the output of a command in a file for further processing.

Example: The following commands will store the output in files rather than sending it to the terminal.

$ cat emp.dat > emp.out <Enter>

will read the file emp.dat and stores the output into the file emp.out.

$ date> todays_date <Enter>

will redirects the output of the date command to a file called todays_date.

$ cat todays_date <Enter>

will display the content of the file todays_date.

$ ls > listing <Enter>

will redirect the output of Is command to a file called listing.

$ cat> newfile <Enter>

In the above command, no input file was given to cat, so it will read the input from the standard input device (the keyboard) and redirect the output to the file newfile.

The output redirection operator, '>', will create the file specified, if it does not already exist. If the target file already exists, then its contents will be overwritten. To append the new matter to the existing content of the target file, the append operator "»" (double greater than symbol) is used.

$ cat » newfile <Enter>

If the target file does not exist then it will be created by the shell.

.Standard Error Redirection

- Standard error is used to display error messages.
- By default the standard error is assigned to the terminal.
- File description or for standard files:
    - 0 is assigned to the standard input
    - 1 is assigned to the standard output
    - 2 is assigned to the standard error
- $ command 2> err_file

In UNIX it is also possible to redirect the error messages of an invalid command to a different destination other than the standard error file so that the error messages do not jumble with the output.

This is done .by using a file descriptor for the standard error file.

In the examples seen earlier, the input and output redirection operators can also be written as "0<" and "1>". .

The numbers before the redirection operators are optional and implicit.

But while redirecting errors the file descriptor has to be used to avoid confusion between output redirection and error redirection.

Example: Redirecting the standard error.

$ cat temp 2> error_mesg <Enter>

In the above example if the file temp does not exist, the error message, which would be

displayed on the terminal, will go to the file error_mesg.

$ cat emp.dat > newemp 2> emp_err <Enter>

In the above example, if the file emp.dat exists, cat will read it and redirect the output to a file called newemp, if there is any error, the error message will go to the emp_err file. No display be produced on the terminal.

Pipes and Filters

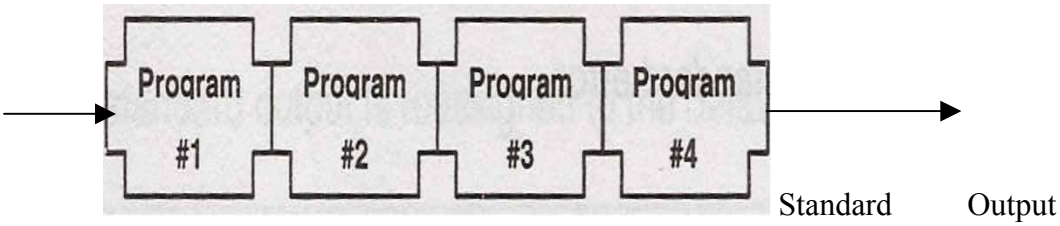
- Connecting two or more commands in tandom: Pipes
- Filters
- Commonly used filters
    - more
    - paste
    - tee
    - head
    - tr
    - tail
    - grep
    - wc
    - sort
    - pg
    - cut

The fact that many UNIX commands accept input from the standard input and send outpul standard output allows us to link them together with a pipe (I).

,With this facility, the shell allows any number of such commands to be connected in a seq

known as a pipeline where the standard output of each command in the sequence is pip the standard input of the next command as shown in the figure.

Standard Input



Programs in a pipeline are executed sequentially. UNIX automatically handles the data fl< one program to the next, producing the effect as if one program is being executed.

A filter is a program or UNIX command that takes its input from standard input process filters) it and sends its output to the standard output. It is obvious that the commands sue date, pwd etc. can not be used as filters as they do not require any input.

Filters can be used anywhere in the pipeline.

.grep

- searching a pattern in a file
- $ grep [options] "<pattern>" <filename>
- options:
    - -n: prints line numbers
    - -v: the reverse search criterion
    - -c: display only a count of matching patterns
- regular expressions:

- ″^″ beginning of line
- "$" end of line
- "." any single character
- [...] anyone character from the list

The grep (Global search for Regular Expression and Print) utility, is the standard file searching and selection utility. It is used to search for a particular pattern of characters and display all the lines containing the pattern.

This utility examines the input file line by line for the given pattern. When a match is found, the line IS copied to the standard output file. No output is produced when the desired pattern is not found.

The format of the grep command is:

$ grep [option] <pattern> <filename>

Example:

.Display the line(s) containing the string **MADHU.**

$ grep "MADHU" emp. Dat <Enter>

.Display the line(s) containing the string **ANURAG** preceded by the line number.

$ grep -n "ANURAG" emp.dat <Enter>

.Display the lines of the file excluding the line(s) containing the string **MADHU.**

$ grep -v "MADHU" emp.dat <Enter>

.**Regular Expressions**

The patterns used by the **grep** utility are called regular expressions. The regular expressions are always given in double quotes.

Regular Expressions	Meaning
[]	To specify a pattern which consists of anyone of a set of characters Example "[xyz]" specifies the pattern either "x" or "y" or "z"
[ ] with hyphen	To specify a pattern which consists of anyone of the range of characters specified in the square brackets. Example "[1-3]" specifies the pattern either 1 or 2 or 3. Similarly "[1-9 a-z]" specifies a pattern. of anyone character which can be a number from 1-9 or a lower case alphabet from a to z.
"'character (caret symbol)	This character is used to specify that the pattern following it must occur at the beginning of the line. Example "/\[123]" specifies that the pattern either 1 or 2 or 3 should appear at the beginning of each line to be selected.
[/\characters]	When a caret symbol is specified within the square brackets and is followed by one or more characters or ranges of more characters (indicated by hyphens) then strings containing any of the specified will be excluded.

	Example "[/\123]" specifies that the searched string should not contain 1, 2 or3
Dot (.)	This character is used in conjunction with any other characters and indicates anyone character Example "[123]." specifies the pattern either 1 or 2 or 3 followed by any single character
\ (Backslash)	This character is used in conjunction with the above characters and indicates that grep should ignore the special meaning of the character following it in the regular expression. Example "\[abc\]" specifies the pattern [abc] since the characters and ']' are treated as ordinary characters and not signifying a set or range of characters.
$(dollar)	To specify that the pa.!!1J1 preceding it must occur at the end of each line. Example "[abc]:>Specifies the pattern either a or b or c at the end of each line.

Example:. Searching through the employee file using the grep command

Consider the file employee having the following data:

PATHAK	MANI	2057	CD	El	3	*03/03/56*	1575
BHATIA	NAVEEN	9090	SALES	E2	4	*03/04/67*	9000
YADAV	JOHNNY	8909	MKT	E3	6	*09/12/78*	8500
ALVA	VI JAY	9009	MKT	E6	7	*09/11/53*	29000
MISHRA	M.R.	2103	ADMIN	E3	7	*03/11/54*	12000
JOSE	JOBBY	9001	MKT	E2	5	*03/11/75*	3000

BABY	P.K.	9002	ADMIN	E3	10	*03/12/69*	6000
JAMEEL	*S.*	9012	CD	E3	10	*03/12/62*	6500
LALITHA	N.	9023	CD	E2	10	*03/12/56*	6400
SINGH	ROHIT	9015	TRG	E2	4	*09/07/69*	6899
SINHA	RAJEEV	1289	CD	E3	11	12/11/62	9812
GULATI	RAJENDER	1190	ADMIN	El	3	*05/03/66*	5400
DUGGAL	TARUNA	1193	ADMIN	E3	7	*02/07/65*	7200
VERMA	MOHIT	*2390*	CD	El	2	02/09/67	4389
SAXENA	RAHUL	3476	ADMIN	E6	18	*01/04/55*	18500
MISHRA	J R	7654	MKT	E3	7	*07/07/64*	9065
DAS	H K	*3420*	MKT	El	2	*05/03/68*	5400
MURALI	N	2978	TRG	E4	8	*04/03/61*	9000
FRANCIS	SHAM	*1390*	TRG	E2	5	*07/07/69*	7000

.To display the records of the employee whose last name begins with B

$ grep "^B" employee

. Display the records of all the employees whose basic pay is between 6000 and 6999

$ grep "6...$" employee

. To display records of all employees whose employee code is between 9000 and 9099

$ grep "90[0-9] [0-9]" employee

.To display the records of all the employees whose department is MKT. Store the result in out.dat.

$ grep "MKT" employee> out.dat

.To print the records of the employee whose first nare is VIJAY along with the line numbers.

$ grep -n "VIJAY" employee

.To display all the lines not having the string "MKT", along with the line numbers.

```
$ grep -vn "MKT"       employee
```