Lecture 10

Unix Commands III: I/O Redirection and Pips

Shibo Li

shiboli@cs.fsu.edu

Department of Computer Science
Florida State University

The slides are mainly from Sharanya Jayaraman

Recapt Part I&II

- `clear`, `who` , `whoami`, `pwd`, `ls`
- `cd`, `vim`, `g++`
- `mkdir`, `rmdir`, `cp`, `mv`, `rm`

Some New Unix Commands

- `date` - Displays the current date and time
- `cal` - The calendar command. Displays the current month ina calendar format. Had options to display other formats.
- `man` - The manual command. Will display the manual for theUnix command given as the parameter.
- `du` - The disk usage command. Shows the number of disk blocks - 512 byte units, being used by the contents of thecurrent directory. Has several options as well.

Some New Unix Commands

- `head` Prints a few lines from the beginning of a file.
- `tail` - Prints a few lines from the beginning of a file.
- `sort` - Sorts the input according to the given option.

Unix Input and Output

- Unix is a command-line based operating system where all interactions between the user and the system happens through text.
- These interactions are usually processed through intermediary system files.

Unix Input and Output

- **Standard Input - stdin:**
  - The default place where a process reads its input
  - Usually the keyboard input
- **Standard Output - stdout:**
  - The default place where a process writes its output
  - Usually, the terminal display on the monitor.
- **Standard Error - stderr:**
  - The default place where a process can send its error messages
  - Usually, also the terminal display on the monitor.

Redirecting Standard I/O

- Standard input and output can be redirected providing a greatdeal of exibility in combining programs and unix tools
- To redirect input from a file, use <
  - `./a.out < input`
  - Any use of `stdin` will instead use `input` in this example
- To redirect output into a file, use >
  - `./a.out > output`
  - `cal > todaysCal`

Redirecting Standard I/O

- Both input and output can be redirected at the same time.
    - `./a.out < input > output`
    - The input to `a.out` will come from `input` instead of the keyboard, and the output from `a.out` will go to `output` instead of the terminal display.
- We can also redirect `stderr` and/or `stdout` at the same time

Appending to a file

- The >> operator Appends to a file rather than redirecting output to the file.
  - `./prog1 > output`
  - `./prog2 >> output`
- The previous line will add the `output` of `prog2` at the end of the output for `prog1`, both in the same file - `output`

Pipes

- Pipes allow the standard output of one program to be used as the standard input of another program
- The pipe operator ‘|’ takes the output from the command on the left and feeds it as standard input to the command at the right of the pipe
- Pipes are more efficient as compared to using intermediate files

Pipes

- Examples
  - `ls | wc -w` Displays the number of files instead of the filenames.
  - `prog1 < input.txt | prog2 | prog3 > output` The contents of input.txt are used as input to prog1. The ouput is used as input for prog2, whose output is sent to prog3 as input. The final output of prog3 is stored in output.txt

Another Pipe Example

```
du -sc * | sort -n | tail
```

- The `du` command is for disk usage (default is in blocks of 512 bytes). The s and c flags are for summarize and give a grand total respectively
- The `sort -n` command will sort by numeric value
- The `tail` command prints out the last few lines of a file

Separating Commands

- We can run multiple commands on one line, separating them with semi-colons.
- Example:

```
ls -l; cal; date
```

- Suppose you need to continue a command to the next line - use the '\' to do so and then continue your command on the next line

```
cat filename | sort \
| wc
```

date

- The date command is used to display the current date and time.
- The optional “format string” allows us to format the date according to our specifications. Some examples:

```
> date
Sun Sep 22 04:24:58 PM EDT 2024
```

```
> date +%Y-%m-%d
2024-09-22
```

```
> date +%m/%d/%y
09/22/24
```

date

```
> date +%j
266 // day of the year
```

```
> date +"%H:%M, %B %d-%y"
16:48, September 22-24
```

The Unix Timestamp

- Unix measure time in the Unix Timestamp - the number of seconds elapsed since midnight on January 01, 1970 UTC.
- Most programming languages and servers use the Unix time as the default time - something akin to the universal truth.
- The date command can be used to generate the Unix timestamp with the +%s option

```
> date +%s
1727038991
```

The `cal` command

- The `cal` command is used to display the current month in the calendar format on the terminal.
- `cal`
- The command also has the following options:
  - For multiple months starting with the current month:
    `cal -n number_of_months`
  - For displaying the week numbers (1-52):
    `cal -w`
  - For displaying the day numbers (1-365):
    `cal -j`
  - For displaying the calendar month for a particular year:
    `cal day month year`

The `cal` command

- The `cal` command is used to display the current month in the calendar format on the terminal.
- `cal`
- The command also has the following options:
  - For multiple months starting with the current month:
    `cal -n number_of_months`
  - For displaying the week numbers (1-52):
    `cal -w`
  - For displaying the day numbers (1-365):
    `cal -j`
  - For displaying the calendar month for a particular year:
    `cal day month year`

Disk Usage - the `du` command

- The du command is used to check the disk usage (in diskblocks) - the amount of space occupied by files and directories.
- Syntax: `du [options] [directory]`
- Some of the options
  - `du directory`: Disk usage of a particular directory
  - `du -h`: Prints the disk usage in human readable form(bytes/kilobytes/megabytes) instead of disk blocks

Disk Usage - the `du` command

- Some of the options
  - `du -s`: Prints the summary (a grand total) of the disk spaceused by the directory
  - `du -a`: Prints the disk usage of ALL the files/directories in thegiven directory.
  - `du -c`: Prints the grand total of the disk usage at the endafter listing each individual file/directory.
  - `du [options] --exclude`: Can exclude the disk usage of certain specified files/directories

The `head` command

- The `head` command is used to print the first few lines (usually 10 lines) of the given file to standard output.
- Syntax: `head [options] [files]`
- Some options:
  - `head -n number filename`: : changes the number of lines printed from the default 10 to the given number
  - `head -c number filename`: Prints only the first “number”bytes from the file.
  - `head file1 file2 ...`: head can be used for multiple files. Each file’s name is printed as a header before the head output
  - `head -q file file2 ...`: Same as above, except the header with the file names are suppressed.

The tail command

- The tail command is used to print the last few lines (usually10 lines) of the given file to standard output.
- Syntax: `tail [options] [files]`
- Some options:
  - `tail -n number filename`: changes the number of lines printed from the default 10 to the given number
  - `tail -c number filename`: Prints only the last “number”bytes from the file
  - `tail file1 file2 ...`: tail can be used for multiple files.Each file’s name is printed as a header before the tail output
  - `tail -f filename`: Watches for changes to the file. Prints the last few lines every time the file changes. Commonly used to watch log files in real time.

The `sort` command

- The `sort` command is used to sort the contents of the given file.
- The sorting is done in **lexicographic** order - where characters are ordered by their ASCII value. In lexicographic order, spaces come first, then numbers(0-9), uppercase characters (A-z) and then lowercase characters (a-z).
- Syntax: `sort [options] filename`

The `sort` command

- Some options:
  - `sort -r filename`: Sorts in reverse order
  - `sort -n filename`: Sorts numerical content as numbers instead of lexicographic order
  - `sort -f filename`: Ignores case while sorting
  - `sort -c filename`: Checks if a file is already sorted. Prints the lines that are not in order. The file is already sorted ifthere is no output.
  - `sort -u filename`: Sorts the file contents and removes the duplicates
- The original file is not changed unless specified