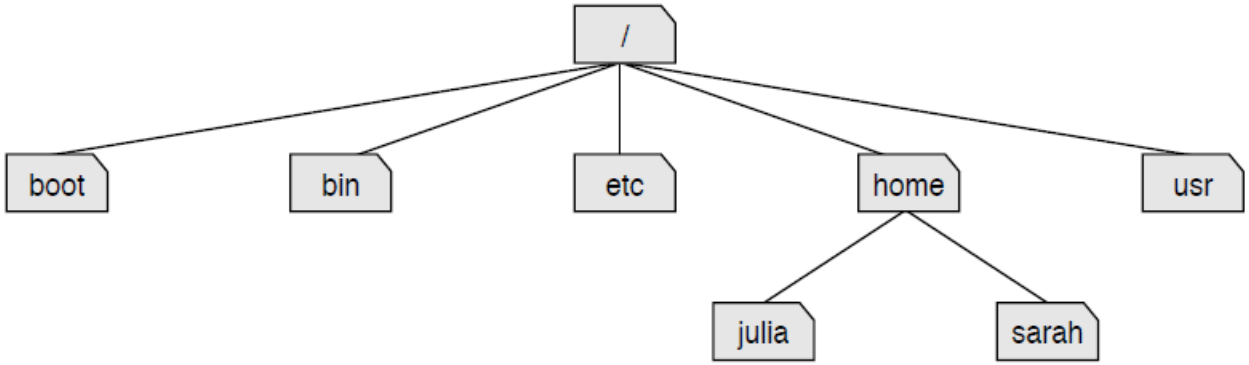
Linux Essentials

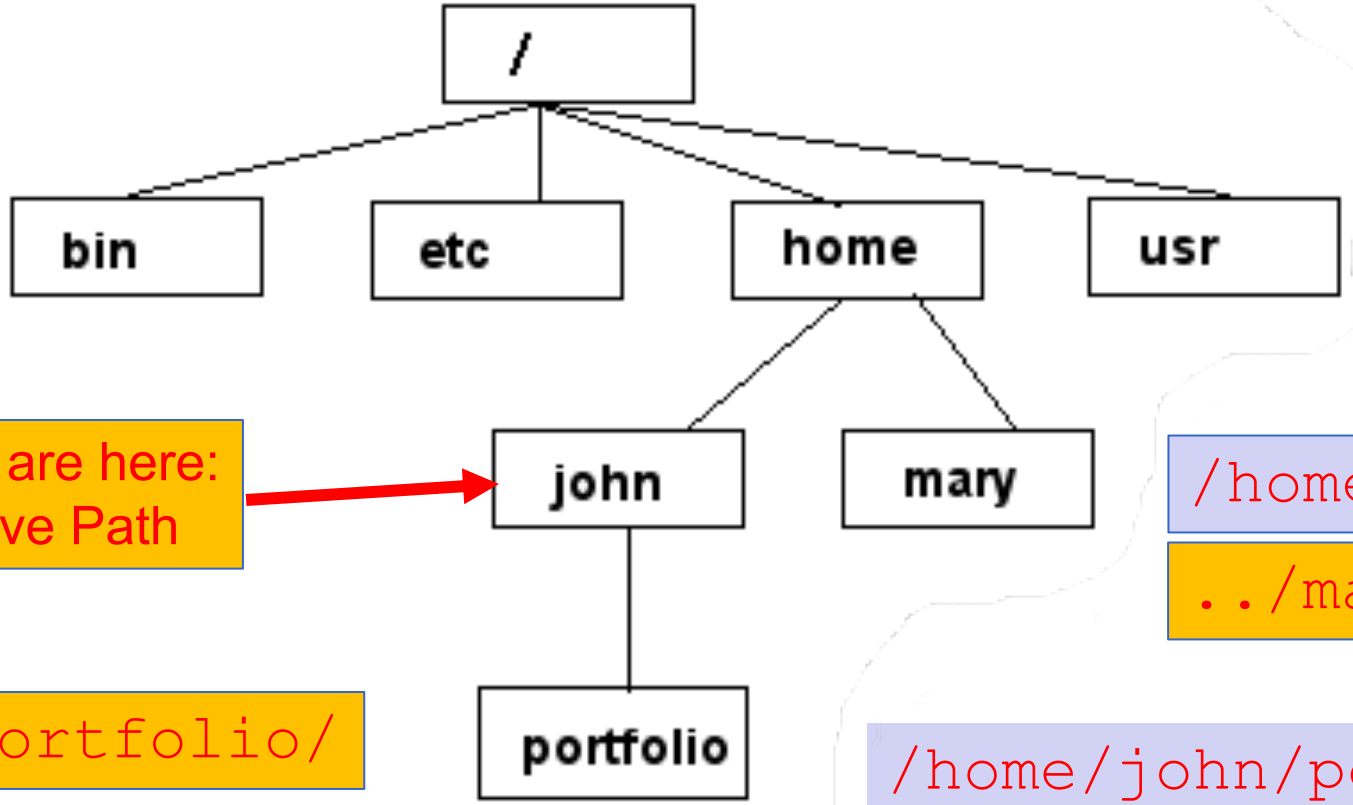
File Management

- Every storage location is accessible under the top-level directory (root)
- The root directory is symbolized by /



Unix/Linux File System

NOTE: Unix file names are **CASE SENSITIVE!**



If you are here:
Relative Path

`./portfolio/`

`/home/mary/`

`../mary/`

`/home/john/portfolio/`

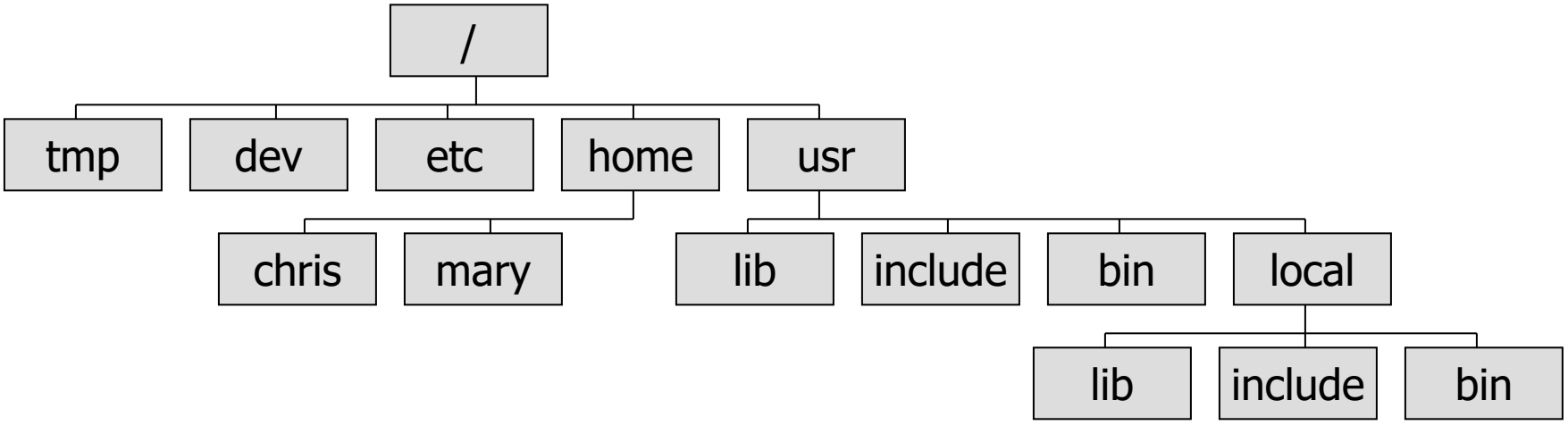
Absolute Path

Filesystem Hierarchy Standard (FHS)

Location	Description/Contents
/	The root or top-level directory.
/bin	Critical binary executables.
/boot	Files related to booting (starting) the system.
/etc	Configuration files for the system.
/home	Regular user home directories.
/lib	Critical system libraries.
/media	Location of mount points for removable media.
/mnt	Location for temporary mounts.
/opt	Optional software packages.
/proc	Information related to kernel data and process data. (This is a virtual filesystem, not a disk-based filesystem.)
/root	Home directory for the root user account.
/sbin	Critical system binary executables.
/tmp	Location for temporary files.
/usr	Location for many subdirectories that contain binary executables, libraries, and documentation.
/usr/bin	Nonessential binary executables.
/usr/lib	Libraries for the executables in the **/usr/bin** directory.

Filesystem Hierarchy Standard (cont.)

Location	Description/Contents
/usr/sbin	Nonessential system binary executables.
/usr/share	Data that is architecture independent.
/var	Data that is variable (changes in size regularly).
/var/mail	Mail logs.
/var/log	Spool data (such as print spools).
/var/tmp	Temporary files.



Intro to Unix: Essential Cmds

cd	- change directory	- cd
mkdir	- make a directory	- md
cp	- copy a file	- copy
ls	- list files	- dir
rm	- remove a file	- del
mv	- move a file	- move & ren
grep	- expression searching	
top	- cpu and memory usage	
who/w	- who else is logged in	
man	- read documentation	

• where am I?

– **pwd**

• who is around?

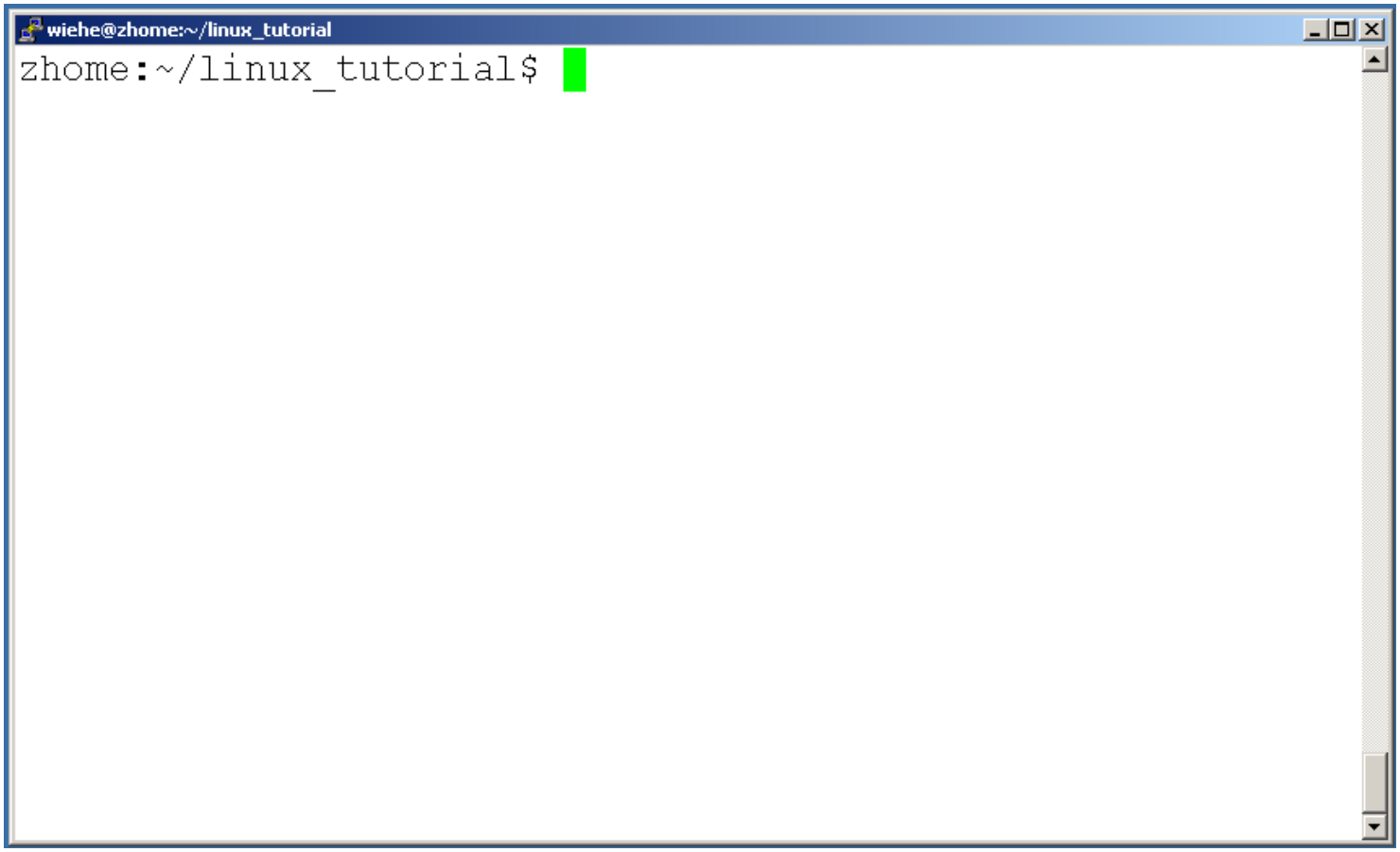
– **who**

• where is that file?

– **find** <path> **-name** <name>

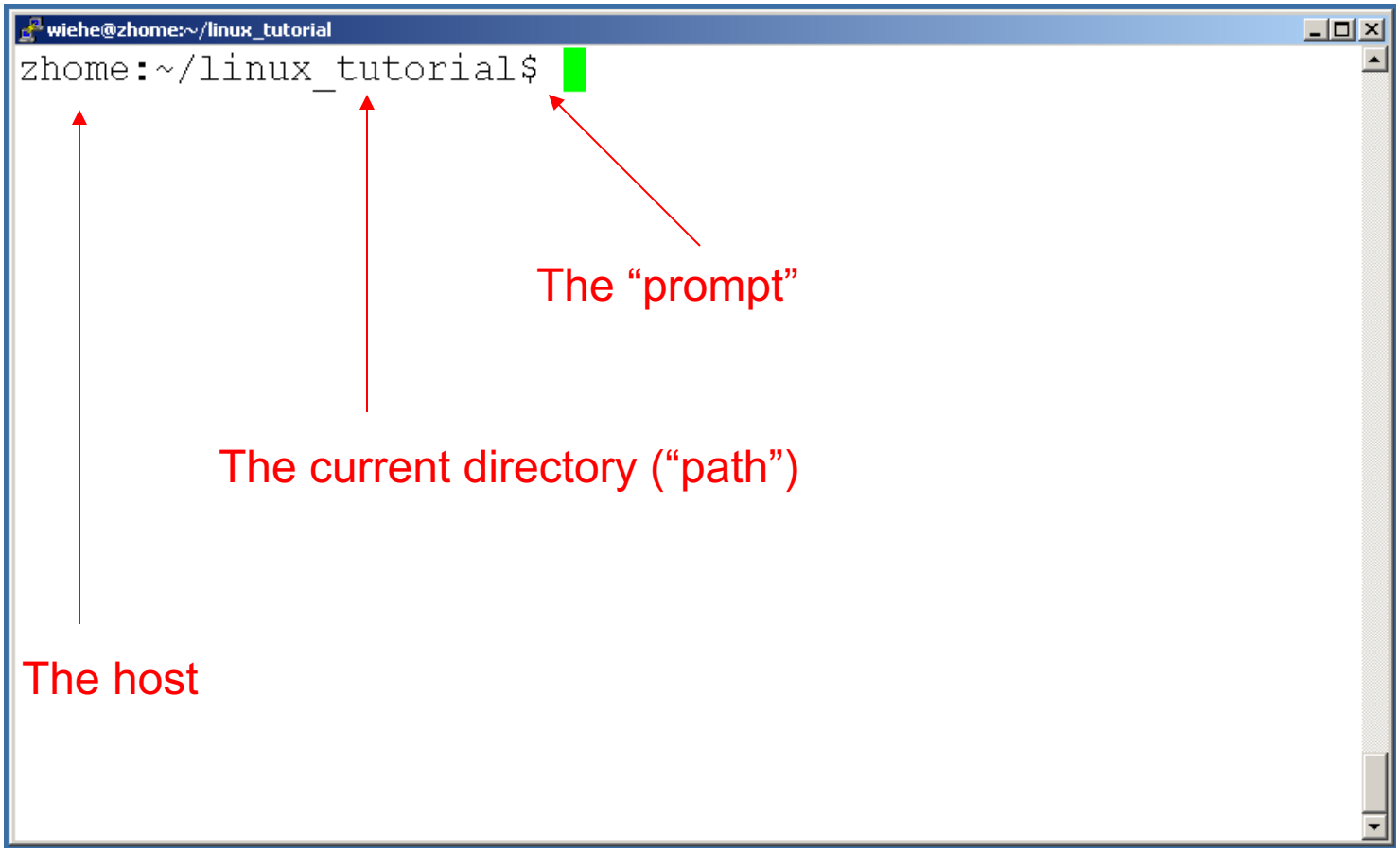
Connecting to a Unix/Linux system

- Open up a terminal:



Connecting to a Unix/Linux system

- Open up a terminal:



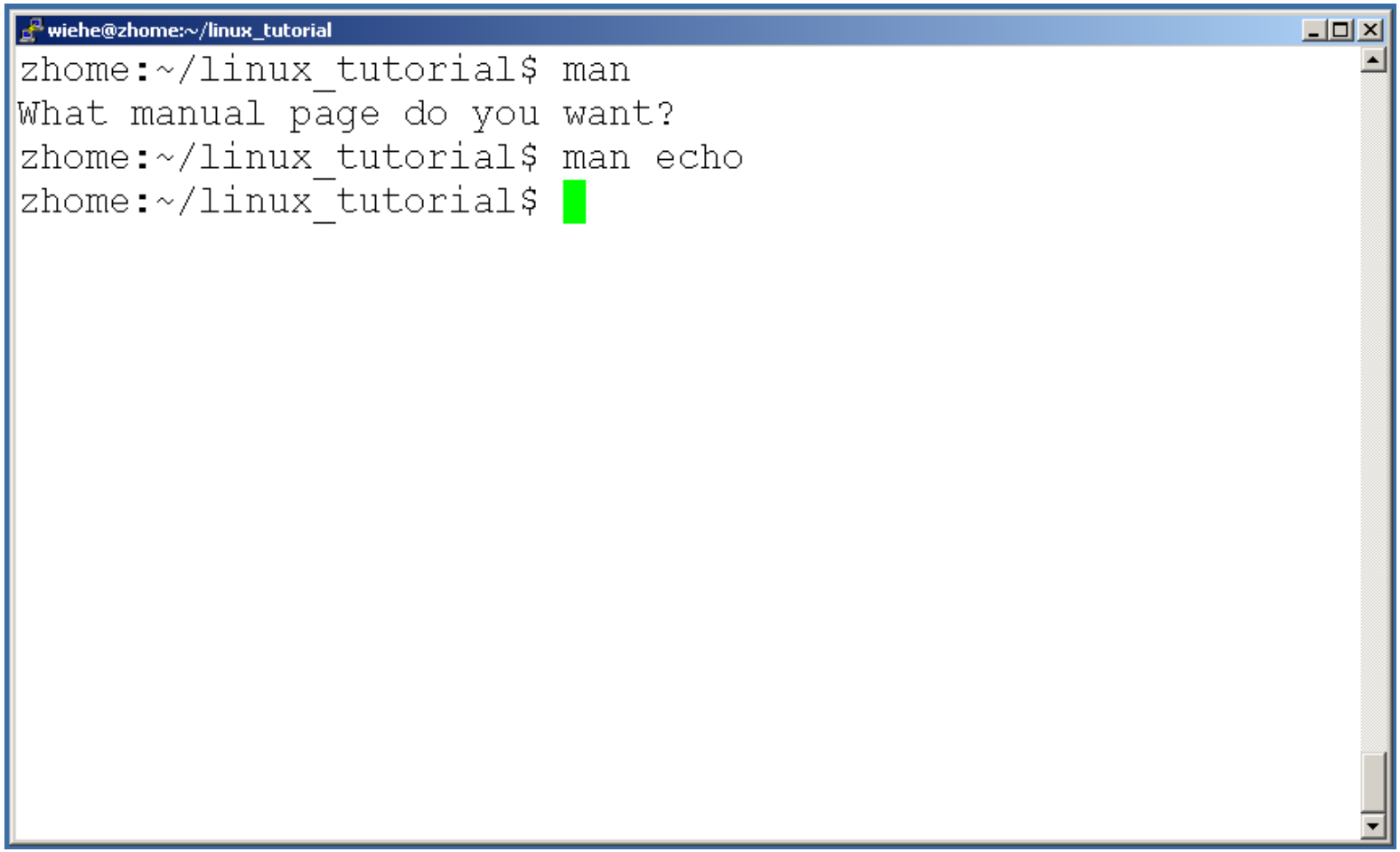
What exactly is a “shell”?

- After logging in, Linux/Unix starts another program called the **shell**
- The shell interprets commands the user types and manages their execution
  - The shell communicates with the internal part of the operating system called the **kernel**
  - The most popular shells are: **`tcsh`**, **`csh`**, **`korn`**, and **`bash`**
  - The differences are most times subtle
  - we are using bash
- Shell commands are **CASE SENSITIVE!**

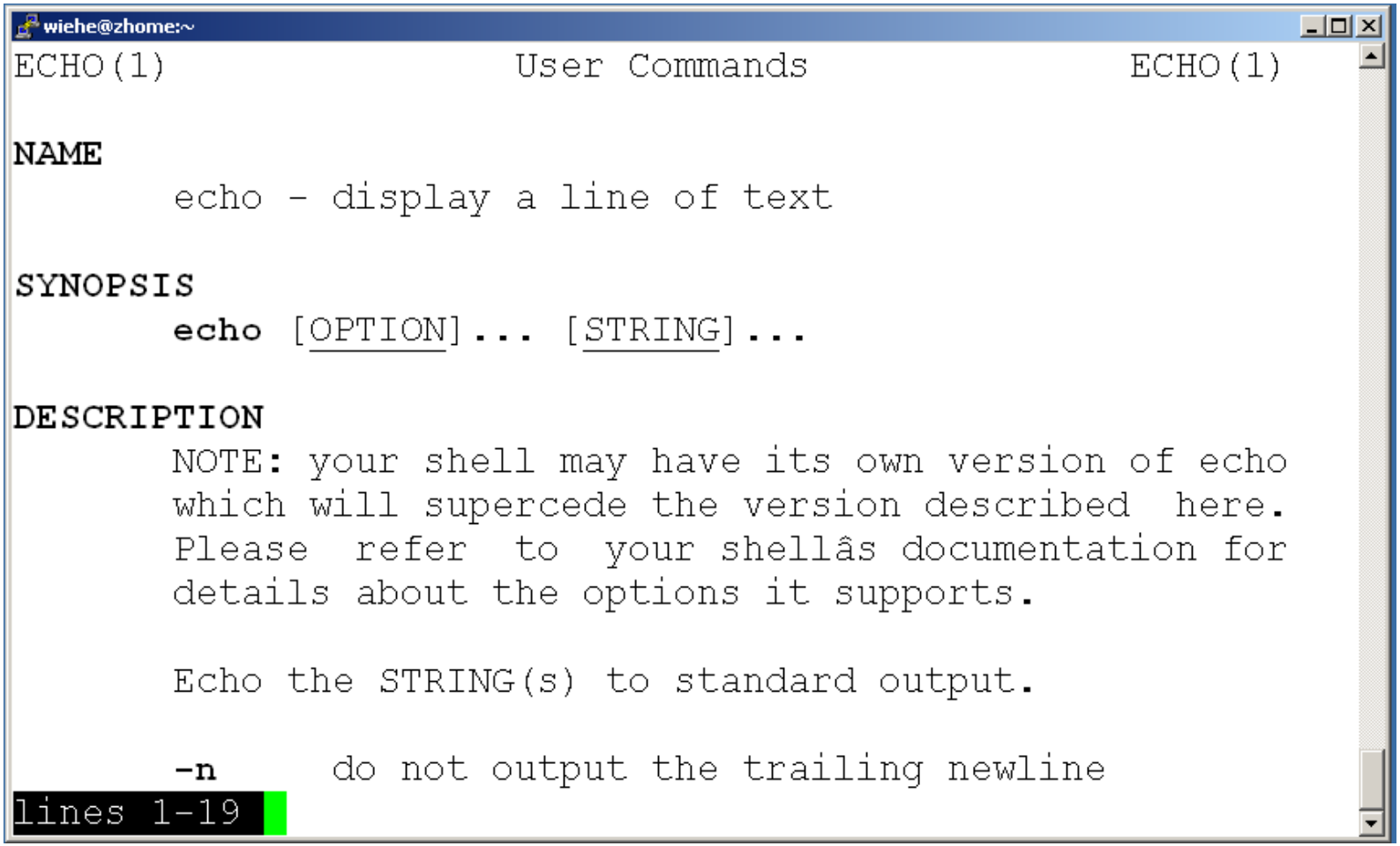
Help!

- Whenever you need *help* with a command type "man" and the command name
- E.g.
  - `$ man ls`
  - `$ man pwd`
  - `$ man gcc`
  - `$ man printf`
  - `$ man malloc`

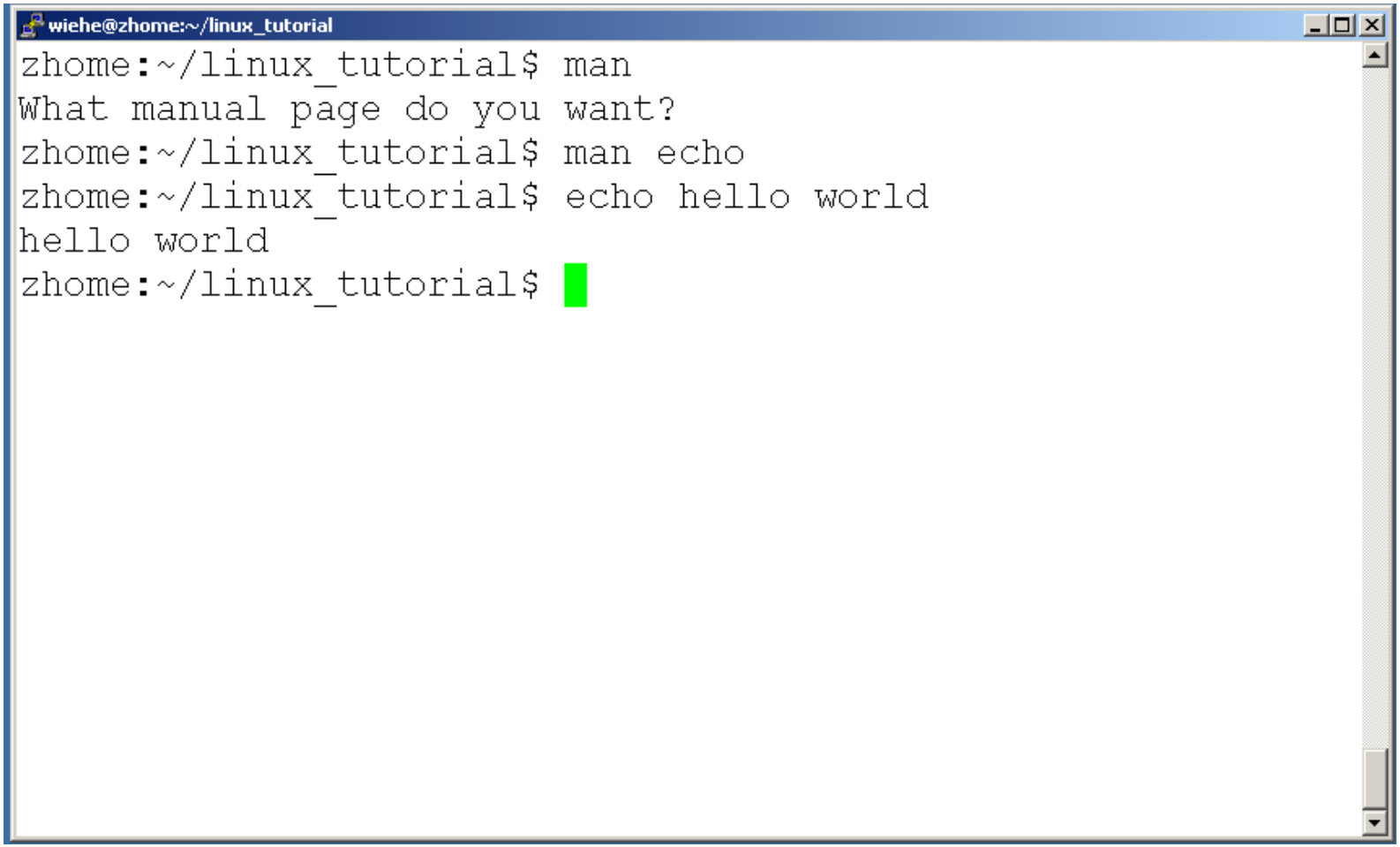
Help!


wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ man
What manual page do you want?
zhome:~/linux_tutorial$ man echo
zhome:~/linux_tutorial$

Help!


wiehe@zhome:~
ECHO(1) User Commands ECHO(1)
NAME
echo - display a line of text
SYNOPSIS
echo [OPTION]... [STRING]...
DESCRIPTION
NOTE: your shell may have its own version of echo
which will supercede the version described here.
Please refer to your shellâs documentation for
details about the options it supports.
Echo the STRING(s) to standard output.
-n do not output the trailing newline
lines 1-19

Help!


wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ man
What manual page do you want?
zhome:~/linux_tutorial$ man echo
zhome:~/linux_tutorial$ echo hello world
hello world
zhome:~/linux_tutorial$

Command Execution

- Type the command and press Enter
- An *option* is a predefined value that changes the behavior of the command
  - Options are a single character value that follows a hyphen, such as –a or –z
  - Some can be combined, as in –az
  - Some newer commands accept word options, such as --long;
    - word options start with *two hyphens*

Command Execution

- An *argument* is a parameter that appears after the command
  - Example: To use the cd command to change to the bin directory: cd /bin
- To execute multiple commands on a single line, separate them with *semi-colons* and press Enter only after the last one
  - Example: pwd ; date ; ls

Common Commands

- pwd
  - Displays the current directory
- cd
  - Changes to another directory
  - Accepts a single *argument*, the desired directory
  - Example: cd /etc
- file
  - Reports the type of content in a file

The ls Command

- ls (*lowercase L, not a capital I*)
  - Lists the files in a directory
  - Default is the current directory; accepts an *argument* of a different directory's name

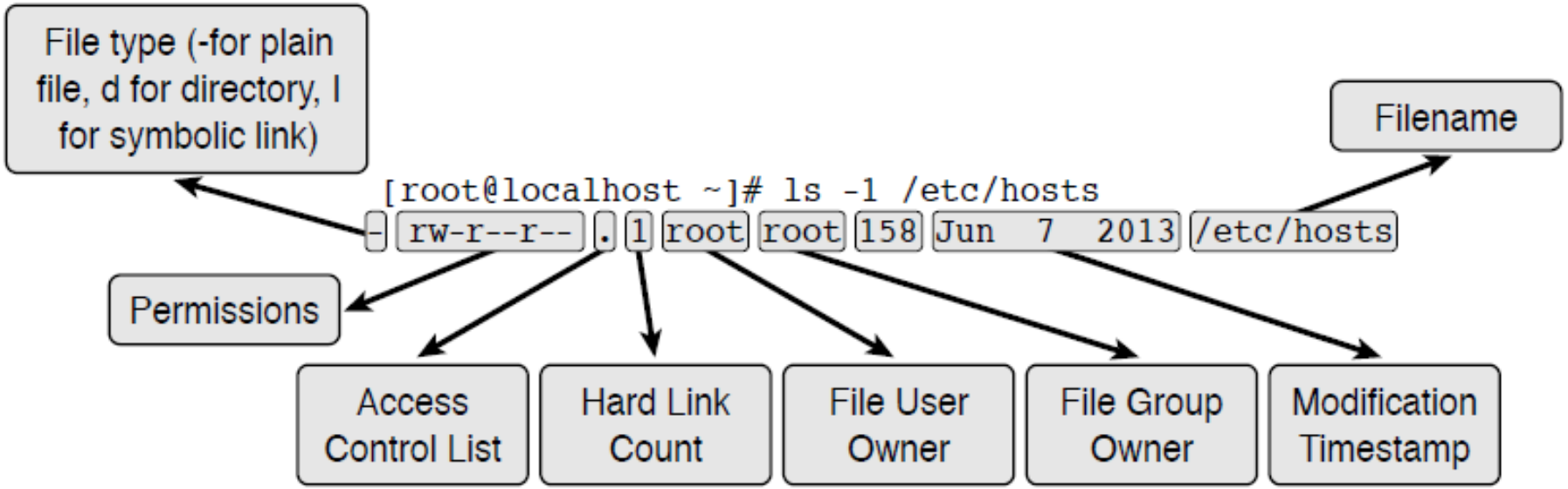
Option	Description
-a	List all files, including hidden files.
-d	List the directory name, not the contents of the directory.
-F	Append a character to the end of the file to indicate its type; examples include * (executable file), / (directory), and @ (symbolic link file).
-h	When used with the **-l** option, file sizes are provided in human-readable format.
-l	Display long listing (see the example after this table).
-r	Reverse the output order of the file listing.
-S	Sort by file size.
-t	Sort by modification time (newest files are listed first).

  - "man ls" for more options

File permissions

- Each file in Unix/Linux has an associated permission level
- This allows the user to prevent others from reading/writing/executing their files or directories
- Use "ls -l *filename*" to find the permission level of that file

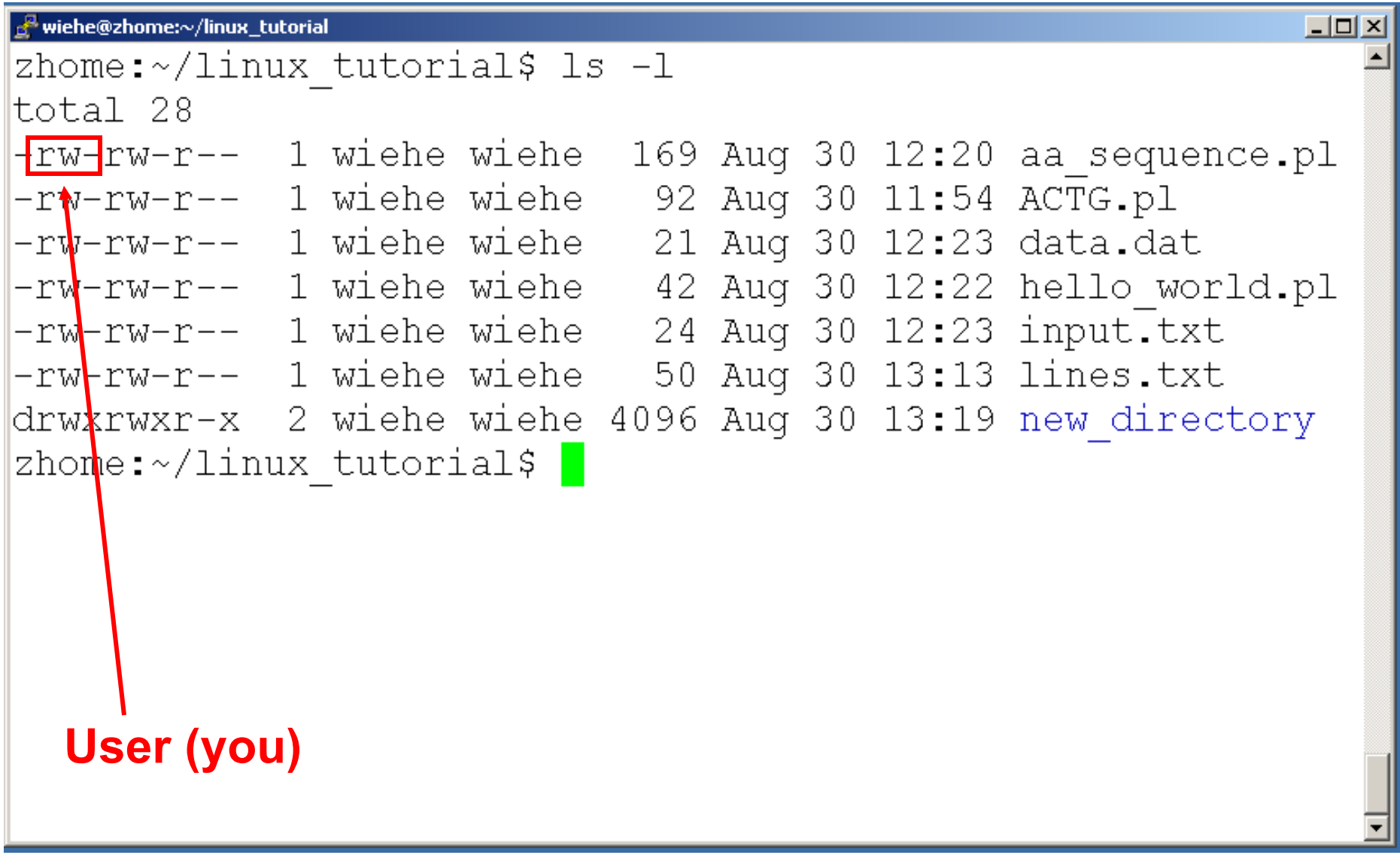
ls Command Output



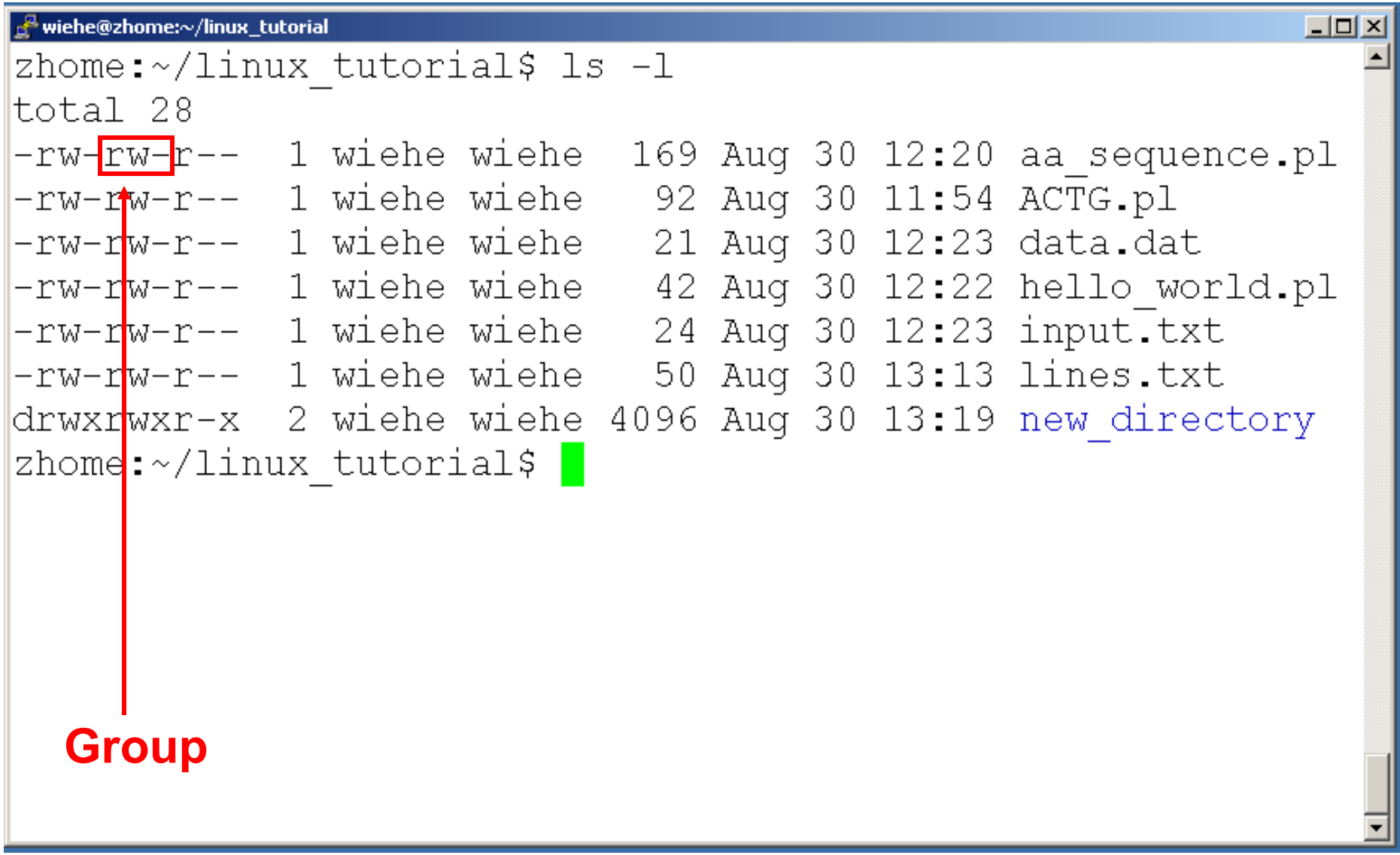
Permission levels

- "r" means "*read only*" permission
- "w" means "*write*" permission
- "x" means "*execute*" permission
  - In case of directory, "**x**" grants permission to list directory contents

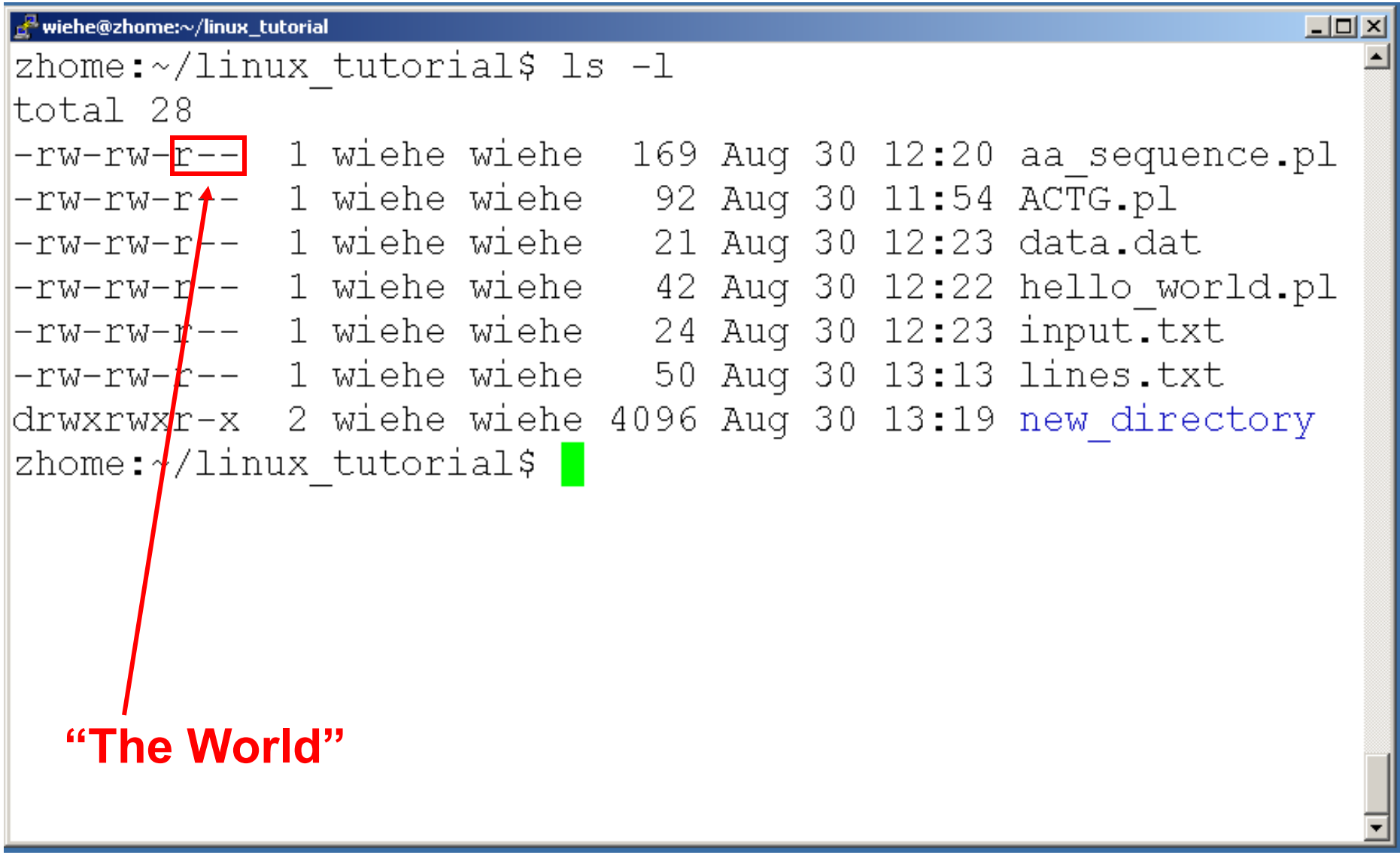
File Permissions



File Permissions



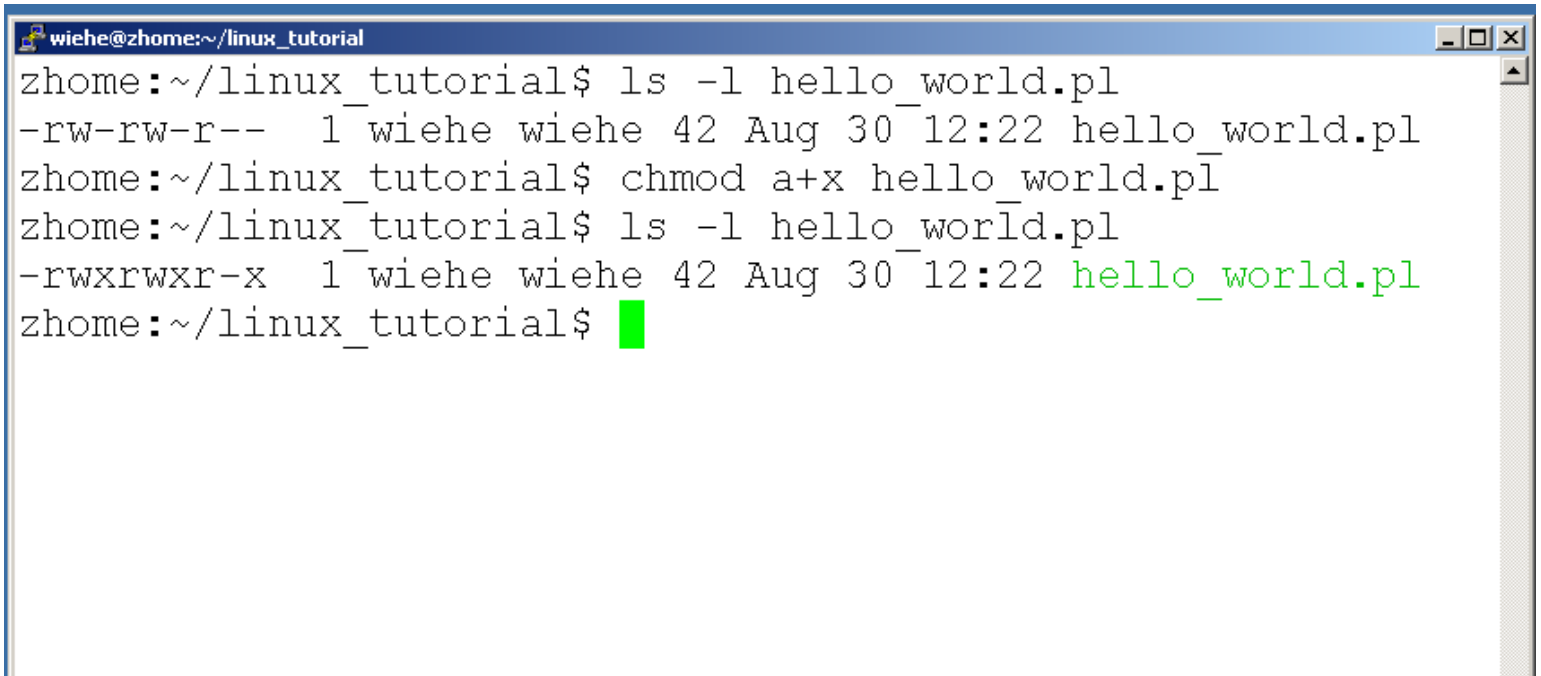
File Permissions



Command: chmod

- If you own the file, you can change it's permissions with "chmod"
  - Syntax:

    $ chmod [**u**ser/**g**roup/**o**thers/**a**ll]+[*permission*] [*file(s)*]
  - Below we grant execute permission to all:



File Globbing

- *File glob (wildcard)*: Any character provided on the command line that represents a portion of the filename

Glob	Description
*	Matches zero or more characters in a filename
?	Matches any single character in a filename
[ ]	Matches a single character in a filename as long as that character is represented within the [ ] characters

• we can use “wildcard”characters to make searches more general

• “*” is the main one, means any set of characthers

• ex:

– find /home/brian -name “*.ppt” : finds all powerpoint files in the account

– grep human *.txt : look for the word “human” in all the files in my directory.

More Common Commands

- less
  - Displays large chunks of text data, pausing after displaying the first page of information
  - Keys on keyboard enable user to scroll through the document
    - Spacebar: one page forward
    - b: one page back
    - Enter or down arrow: down one line
    - Up arrow: up one line
    - q: return to the shell

More Common Commands (cont.)

- head
  - Displays the **top part** of text data
  - By default the *top ten lines* are displayed
- tail
  - Displays the **bottom part** of text data
  - By default the ***bottom ten lines*** are displayed
- mkdir
  - **Creates** (makes) a directory
  - Argument of the new directory's **name**

More Common Commands (cont.)

- cp
  - Copies files or directories
  - Syntax:
    $ `cp [options] file|directory destination`
    - where *file*|*directory* is what to copy and *destination* is where it should be copied
- mv
  - Moves or renames a file
- rm
  - Removes (deletes) files and directories

More Common Commands (cont.)

- rmdir
  - Removes (***deletes***) empty directories
  - Command will fail if not empty unless the –r option is used
- touch
  - **Creates an empty file** if it doesn't exist already
  - Updates the modification and access timestamps if an existing file is specified

Shell Variables

- HOME
- ID
- LOGNAME
- OLDPWD
- PATH
- PS1
- PWD

- Try out: `$ echo $HOME` , or `$ echo $PATH`

echo Command

- Used to display information, such as the value of a variable
- Can also print literal text strings
- Special character sequences:
  - `\a` Ring terminal bell
  - `\n` New line
  - `\t` Tab
  - `\\` Single backslash

set Command

- **Displays** all shell variables and values when no arguments are used
- **Can also modify** the behavior of the shell

Option	Description
-b	When a background job terminates, report this immediately to the shell. A background job is a program running in the background (see Chapter 22, "Connecting to Remote Systems," for details). Use **+b** (the default) to have this report occur before the next primary prompt is displayed.
-n	A shell programming feature that reads commands in the script but does not execute the commands. Useful for syntax-error-checking a script.
-u	Issue an error message when an unset variable is used.
-C	Does not allow overwriting an existing file when using redirection operators, such as ***cmd > file***. See the discussions about redirection later in this chapter for more details on this feature.

- Unset command removes a variable

PS1 Variable

- Defines the primary prompt
- Uses special character sequences
  - `\u` = current user's name
  - `\W` = current directory

PATH Variable

- Contains a comma-separated list of directory locations
- Example:
- `/usr/local/bin:/usr/local/sbin:/usr/bin:/usr/sbin:/bin:/sbin:/home/student/.local/bin:/home/student/bin`
- The path is searched in that order when a command is called
- To execute a command that is not in the path, you must use a fully qualified path name

Environment Variables

- To pass variables and their values to other commands, convert an existing local variable to an environment variable with export
  - Example: export NAME

- If the variable doesn't already exist, export **creates** it directly as an environment variable

- The command *export –p* **displays** all environment variables

The env Command

- Displays environment variables in the current shell
- Local variables are not displayed
- Can also **temporarily set** a variable for the execution of a command
- To unset a variable when executing a command, use `--unset=VAR` where VAR is the variable

Initialization Files

- Logging in starts a login shell
- When a user starts a new shell after login, it is a non-login shell
- Which initialization files are executed depends on whether the shell is a login shell or a non-login shell

Alias

- A shell feature that allows multiple commands to be executed by issuing a single command
- Use alias command with *no arguments* to display all aliases
- Use the unalias command to unset an alias

Command History

- Use history command to view command history
- Limit the number of commands displayed with an argument
  - Example: $ `history 5`

Option	Description
-c	Clear the history list for the current BASH shell.
-r	Read the contents of the history file (see the section "The .bash_history File" in this chapter) and use those contents to replace the history list of the current BASH shell.
-w	Write the history list of the current BASH shell to the history file (see the section "The .bash_history File" in this chapter).

Methods for Redirection

- Standard input: **`stdin`** or **`STDIN`**
- Standard output: **`stdout`** or **`STDOUT`**
- Standard error: **`stderr`** or **`STDERR`**

Method	Description
cmd < file	Override stdin so the input comes from the file specified.
cmd > file	Override stdout so the output goes into the file specified.
cmd 2> file	Override stderr so the output goes into the file specified.
cmd &> file	Override both stdout and stderr so the output goes into the file specified.
cmd1 | cmd2	Override stdout from **cmd1** so it goes into **cmd2** as stdin. See the section "Piping" for more details regarding this feature.

Piping and Subcommands

- Piping
  - Routes the output of a command to another command
  - Uses the "pipe" symbol: |
  - Example:
    - `ls -l /etc | grep "Apr 16"`
    - `ls | grep "mmk"`
    - `ls > directory_listing`
- Subcommands
  - To use one command's output as an argument in another command, place the command within the $( ) characters
  - Example: **`echo "Today is $(date)"`**

Regular Expressions (REs)

- A character or set of characters designed to match other characters
- Examples:
  - A dot (.) matches a single character of any type
  - [a-z] matches any single lowercase character
- Basic and extended REs

Basic REs

RE	Description
^	Match the beginning of a line.
$	Match the end of a line.
*	Match zero or more characters.
.	Match exactly one character.
[ ]	Match exactly one character that is within the [ ] characters; a list of characters ([abc]) or a range of characters ([a-c]) is permitted.
[^]	Match exactly one character that is *not* within the [ ] characters; a list of characters ([^abc]) or a range of characters ([^a-c]) is permitted.
\	Escape the special meaning of a regular expression; for example, the pattern \.* would match the value ".*".

Extended REs

RE	Description
( )	Group sets of characters together to form an expression; for example, **(abc)**.
X|*Y*	Match either *X* or *Y*.
+	Match the preceding character or expression one or more times.
{*X*}	Match the preceding character or expression *X* times.
{*X*,}	Match the preceding character or expression *X* or more times.
{*X*,*Y*}	Match the preceding character or expression *X* to *Y* times.
?	The previous character or expression is optional.

Other Commands

- grep
  - Searches files for lines that contain a specific pattern
  - Example: $ `grep "the" /etc/rsyslog.conf`

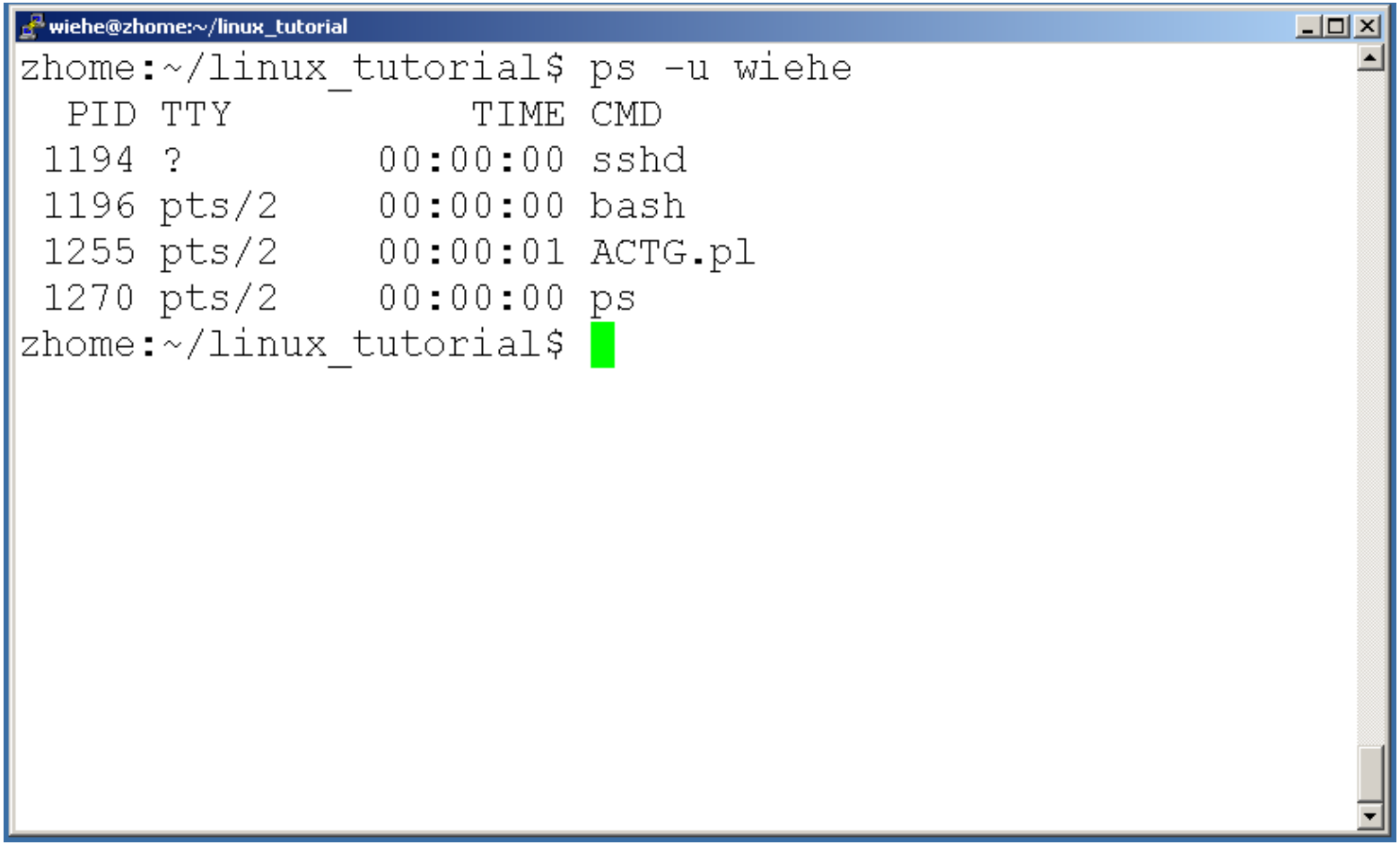
- Sed
  - Edits file data in a non-interactive method (stream editor)
    - `$sed 's/unix/linux/' test.txt`
      - replaces the word “unix” with “linux” in the file.

Compression Commands

- tar
  - Merges multiple files into a single file
- gzip
  - Creates a compressed version of the file
  - Use gunzip to decompress gzipped files
- bzip2
  - Replaces the original file with the compressed file
- xz
  - Compresses files

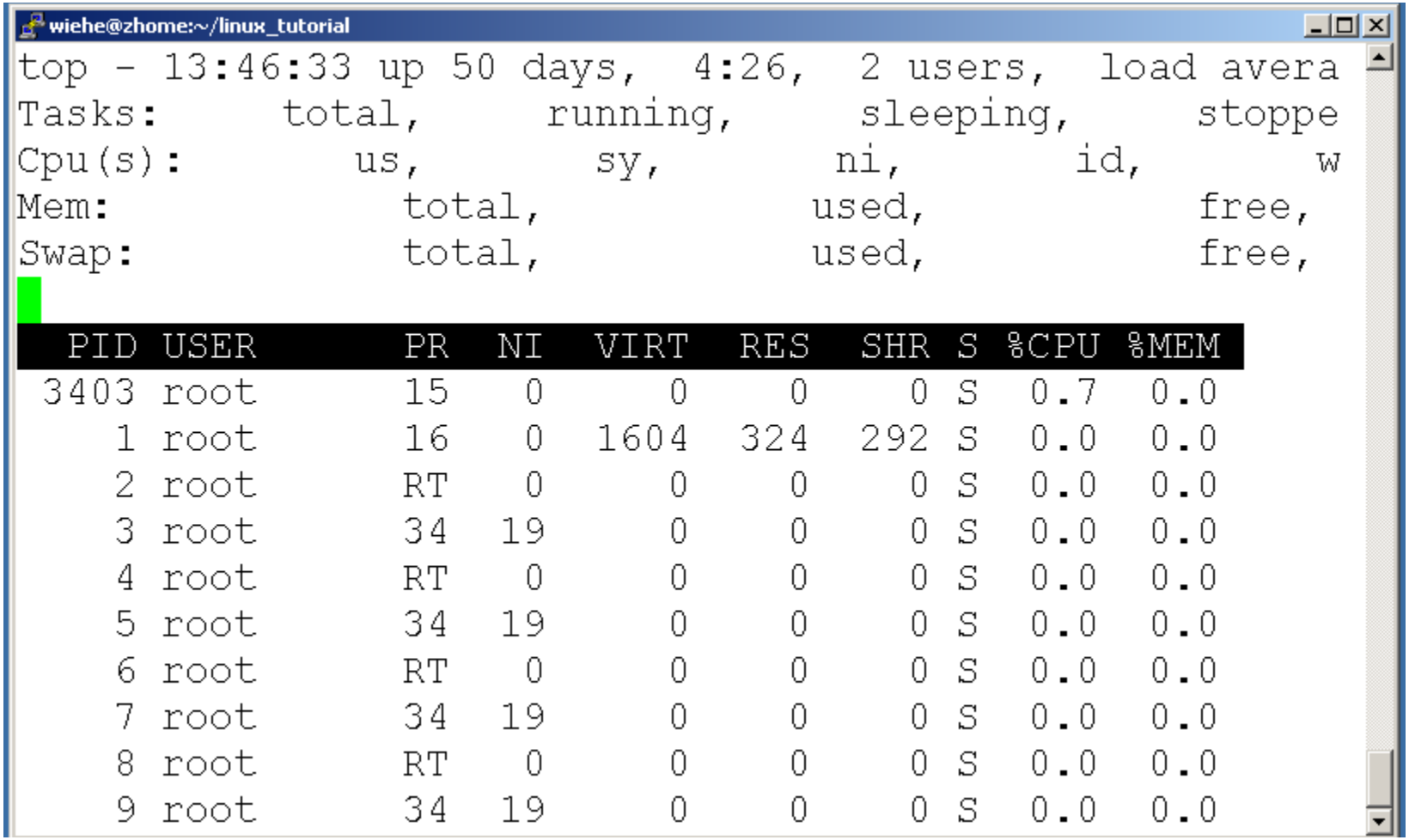
Command: ps

- To view the processes that you're running:



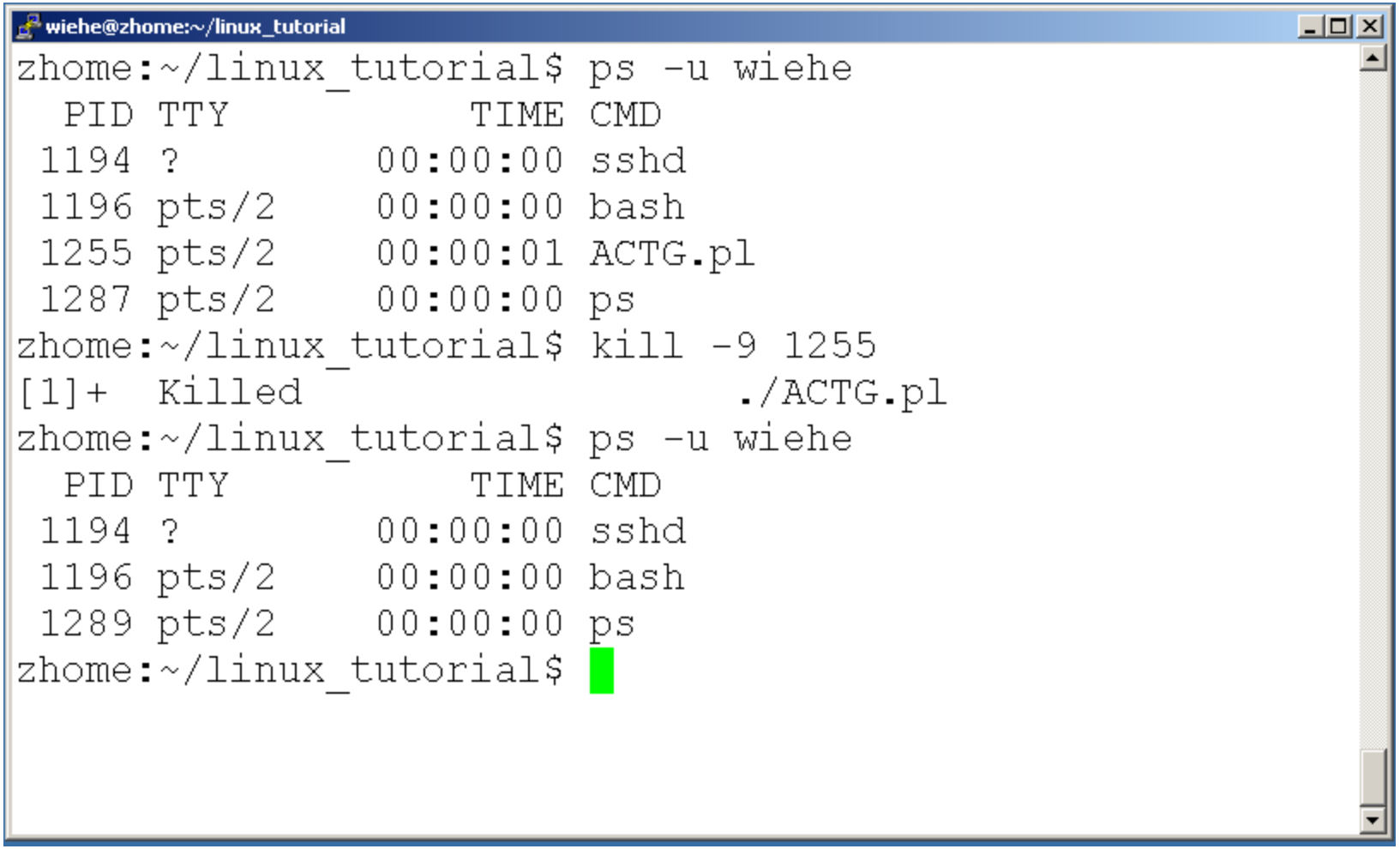
Command: top

- To view the CPU usage of all processes:



Command: kill

- To terminate a process use "kill"



Summary

- This chapter focused on essential commands that all Linux users should know
- These commands enable you to navigate the Linux operating system, manipulate files and directories, and perform advanced end-user operations
- Master these commands before moving on to more advanced security-related topics