



# ASTERISK

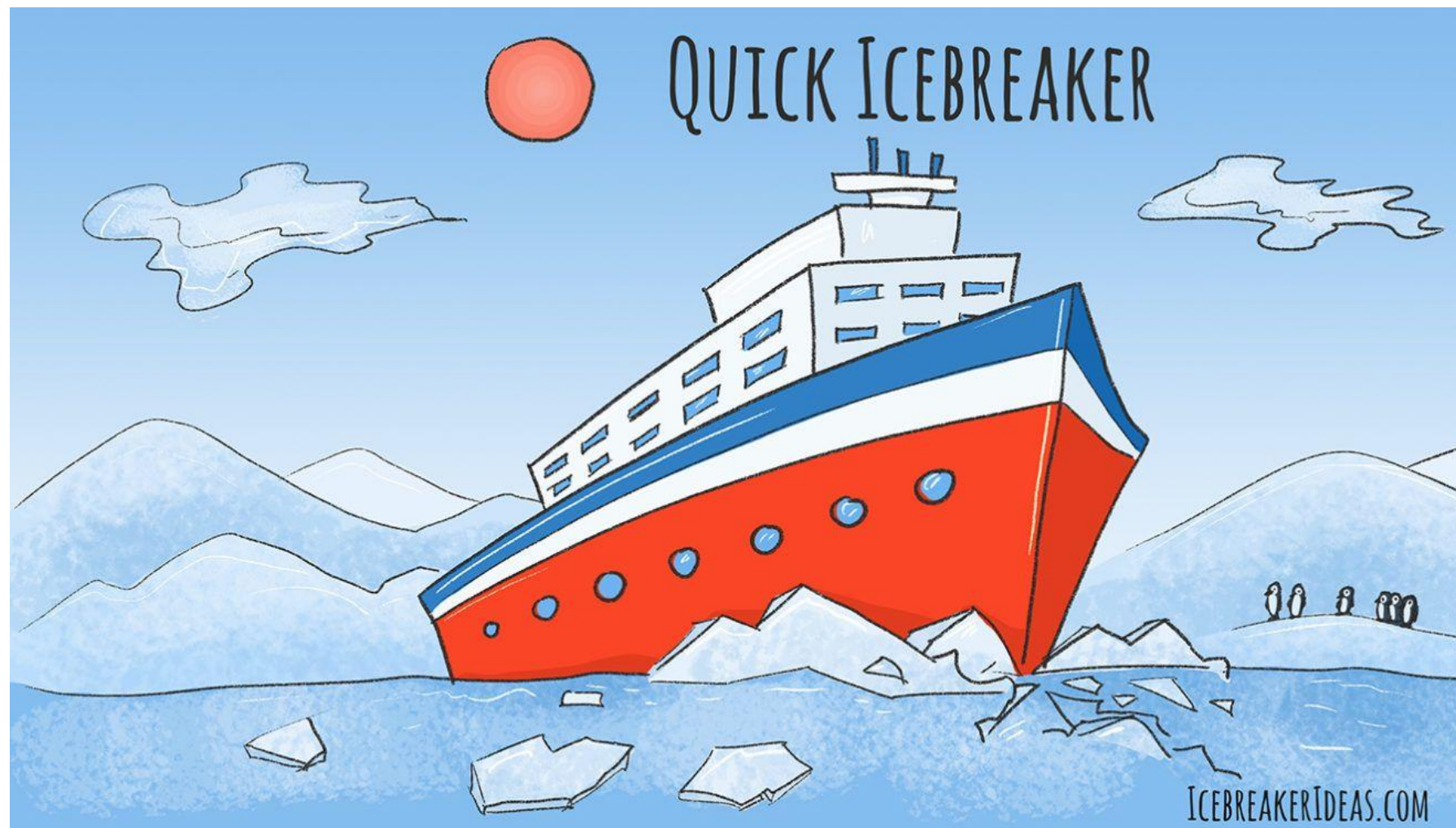
Linux

**TKL** **Teknowlogic**  
The Power of Knowledge

# Contenido

- Tendencias
- Intro a Docker
- Usando imágenes Asterisk para Docker
- Intro a Linux
- Instalando Linux

# Rompamos el hielo...





**Me presento, soy ALFONSO AYALA PALOMA**

## **MAGISTER EN INGENIERÍA – AREA SISTEMAS Y COMPUTACIÓN**

### **RESUMEN – HOJA DE VIDA (PERFIL PROFESIONAL)**

Magíster en Ingeniería en el área de Sistemas y computación de la Universidad Nacional de Colombia. Especialista en Seguridad de la Información de la Universidad de los Andes, Especialista en Docencia Universitaria de la Universidad Cooperativa, Profesional en Ingeniería de Sistemas de la Universidad Nacional de Colombia. Catedrático en las Universidades Cooperativa y del Tolima. Amplia experiencia en proyectos de desarrollo de sistemas de información, herramientas de soporte a toma de decisiones, proyectos Asterisk \* y coaching de Innovación.

# TENDENCIAS

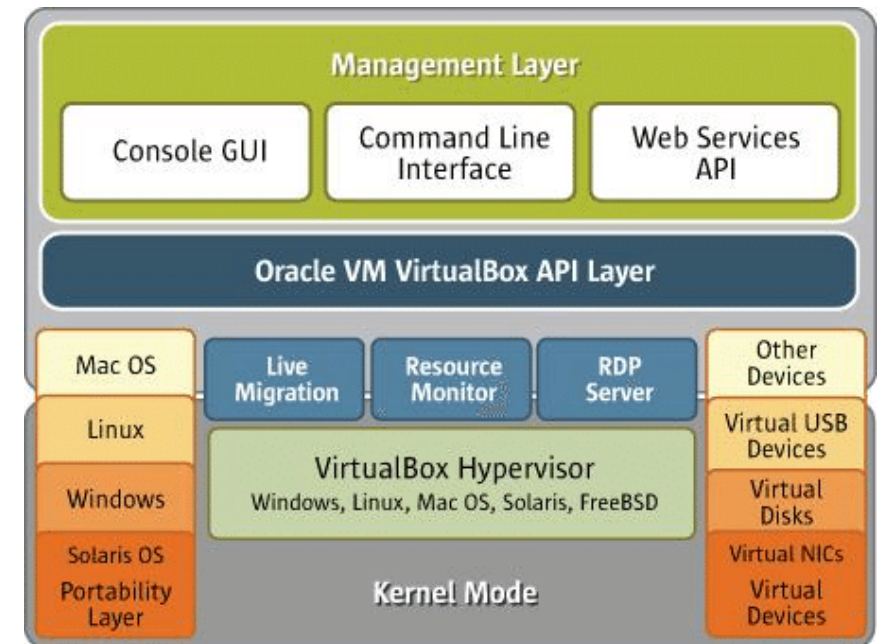
O para donde van las cosas

# Algunas tendencias

- Virtualización
- Contenedores
- Cloud

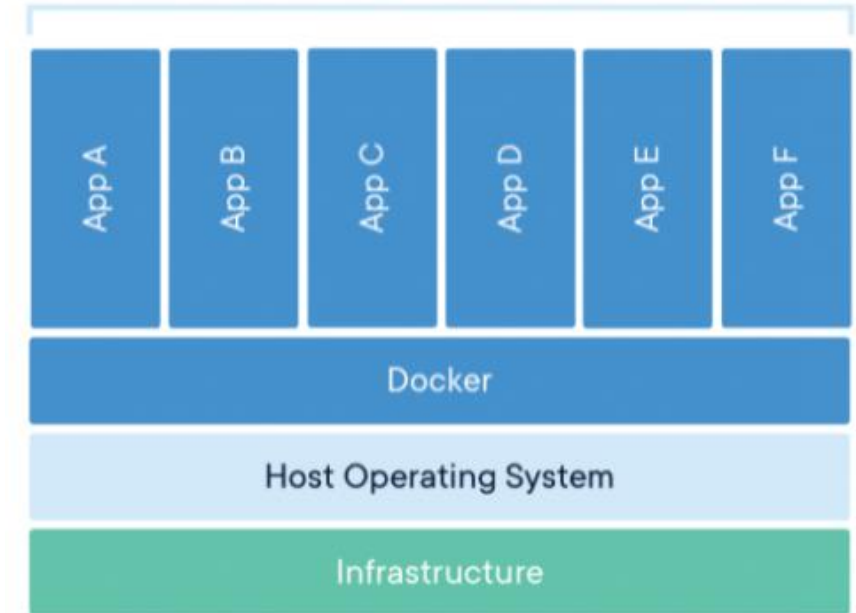
# Virtualización

- Software que crea y corre VM (virtual machines), un hypervisor permite a un computador host soportar múltiples VM guest compartiendo sus recursos como memoria y procesador virtualmente.
- Una maquina simula muchas maquinas



# Contenedores

- Software que permite correr una aplicación incluyendo todos sus prerequisites. Puede intercambiarse entre maquinas
- Una maquina sostiene muchos contenedores

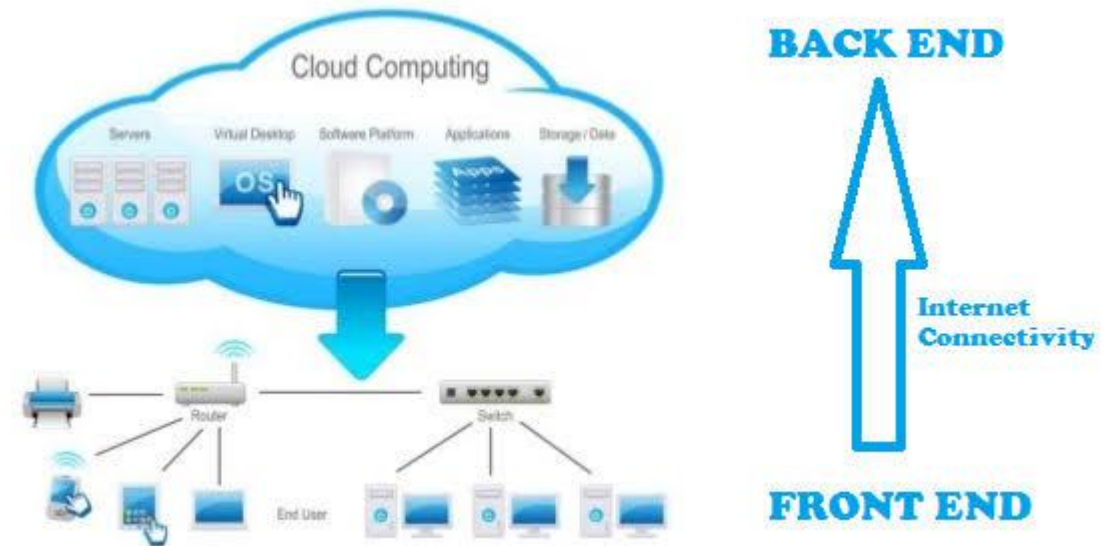




# Cloud

- Localización remota de la infraestructura de cómputo.
- Una nube sostiene muchas maquinas/contenedores

## CLOUD ARCHITECTURE



# Intro a Docker

# Docker

- `docker pull christoofar/asterisk`
- `docker run -p 5060:5060/udp -p 4569:4569/udp --name asterisk christoofar/asterisk`
- `docker exec -it asterisk asterisk -rvvvvv`
- `Docker ps`
- Docker images

# Intro a Linux



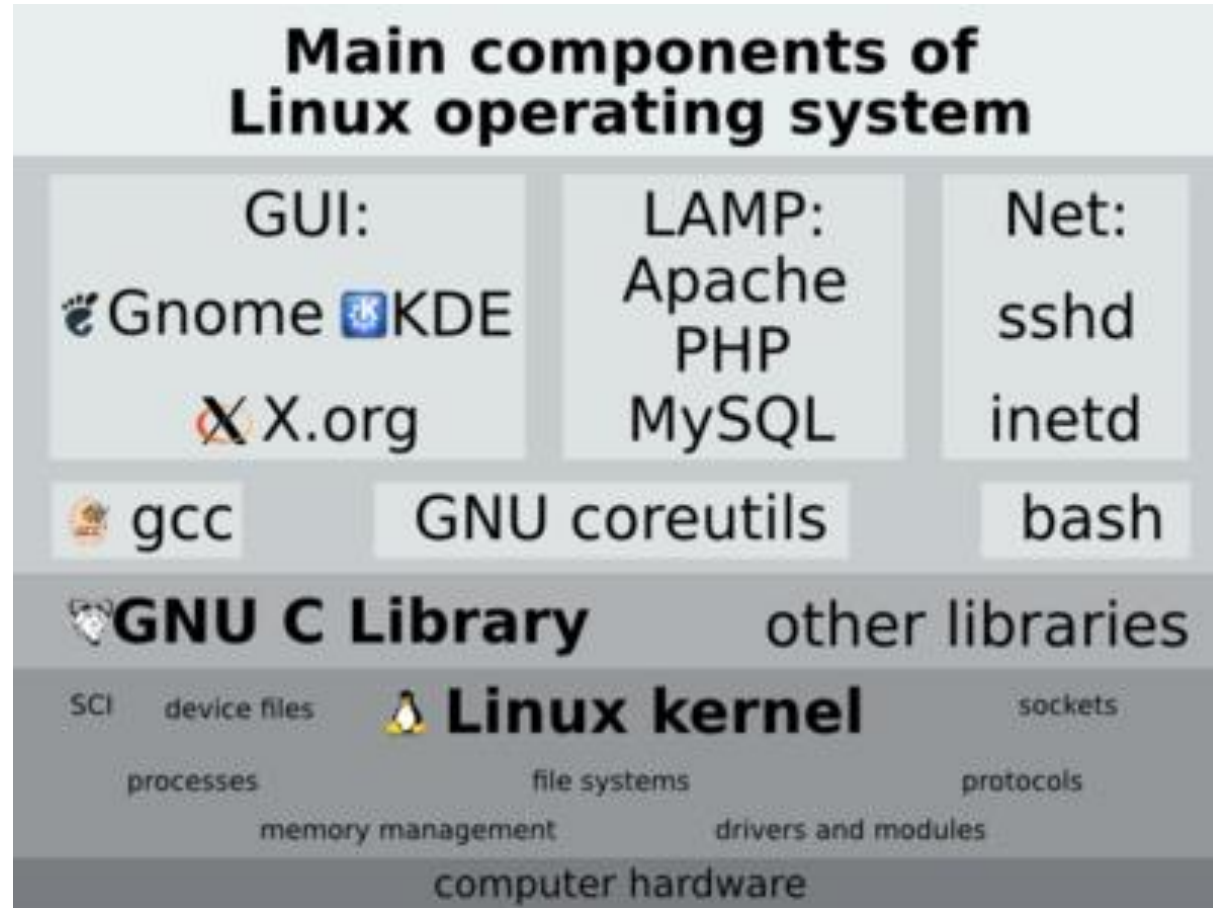
# Linux



# Qué es?

- Sistema Operativo. Android se mueve gracias a Linux.
- Un OS es un software que administra todos los recursos de hardware asociados con la maquina.
- El OS administra la configuración entre el software y el hardware

# Componentes



# Distribuciones





# Clientes SSH

- PUTTY
- MobaXterm

# Linux components

- Bootloader: (GRUB Grand unified bootloader)
- Kernel: manages hardware
- Init System: bootstraps the user space, charges daemons
- Daemons: Background processes/services
- Graphical Server: X
- Desktop Environment: (GNOME, Cinnamon, Mate, Pantheon, Enlightenment, KDE, Xfce, etc.)
- Applications: Other software.

# Linux commands



# Linux test

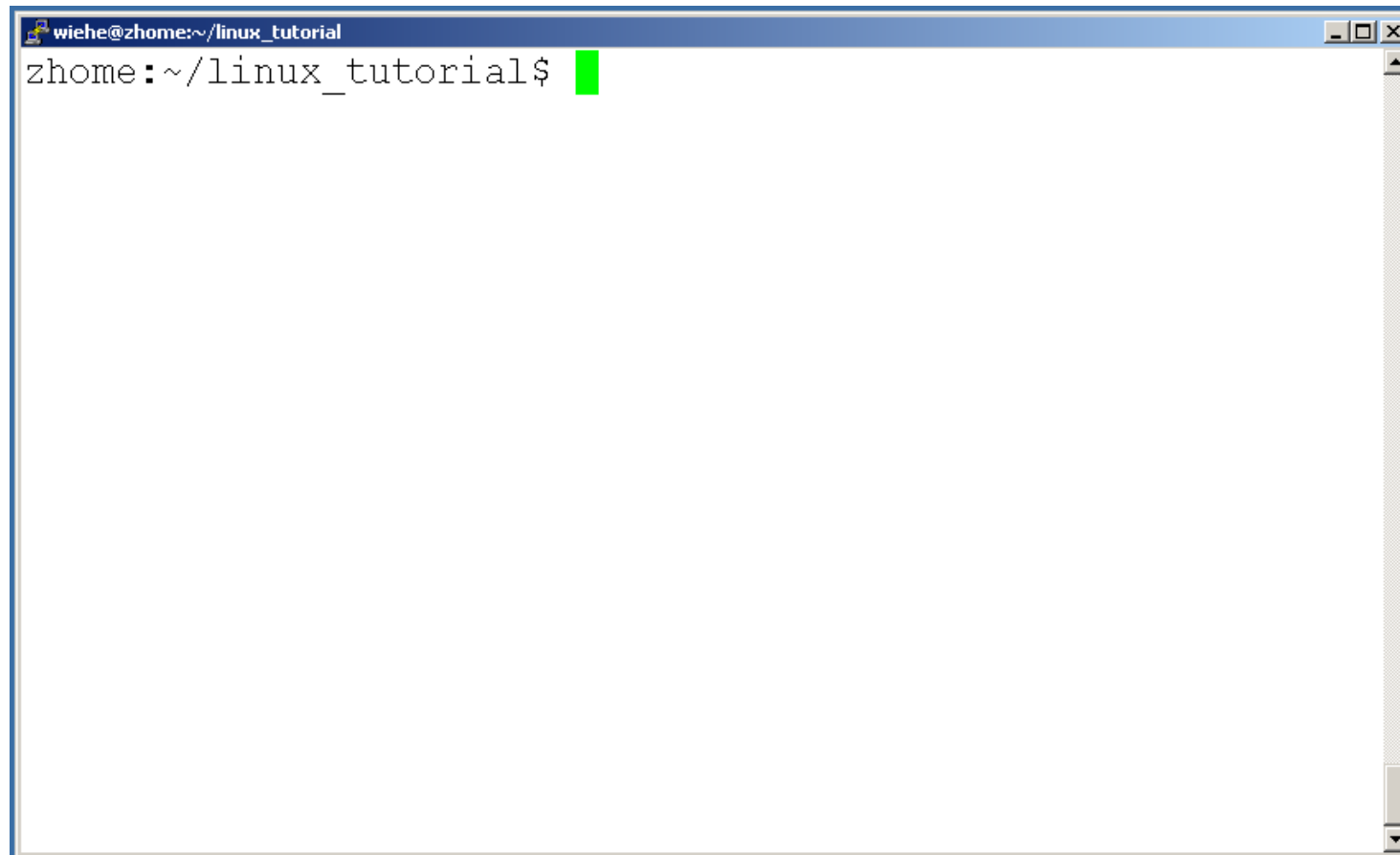
- <https://create.kahoot.it/details/ae07a5f7-e7fc-4e94-958b-5d3d6266a175>

# Linux particularities

- Root : only one who modifies the root directory
- Sudo : superuser
- Ps : process status
- Kill: terminate a process
- Tar: tape archive
- Make: build from sourcecode
- Yum: Yellowdog updater modified (uses RPMs)
- RPM: RPM Package Manager (recursive acronym)
- Pipe: Connect a process output with another input

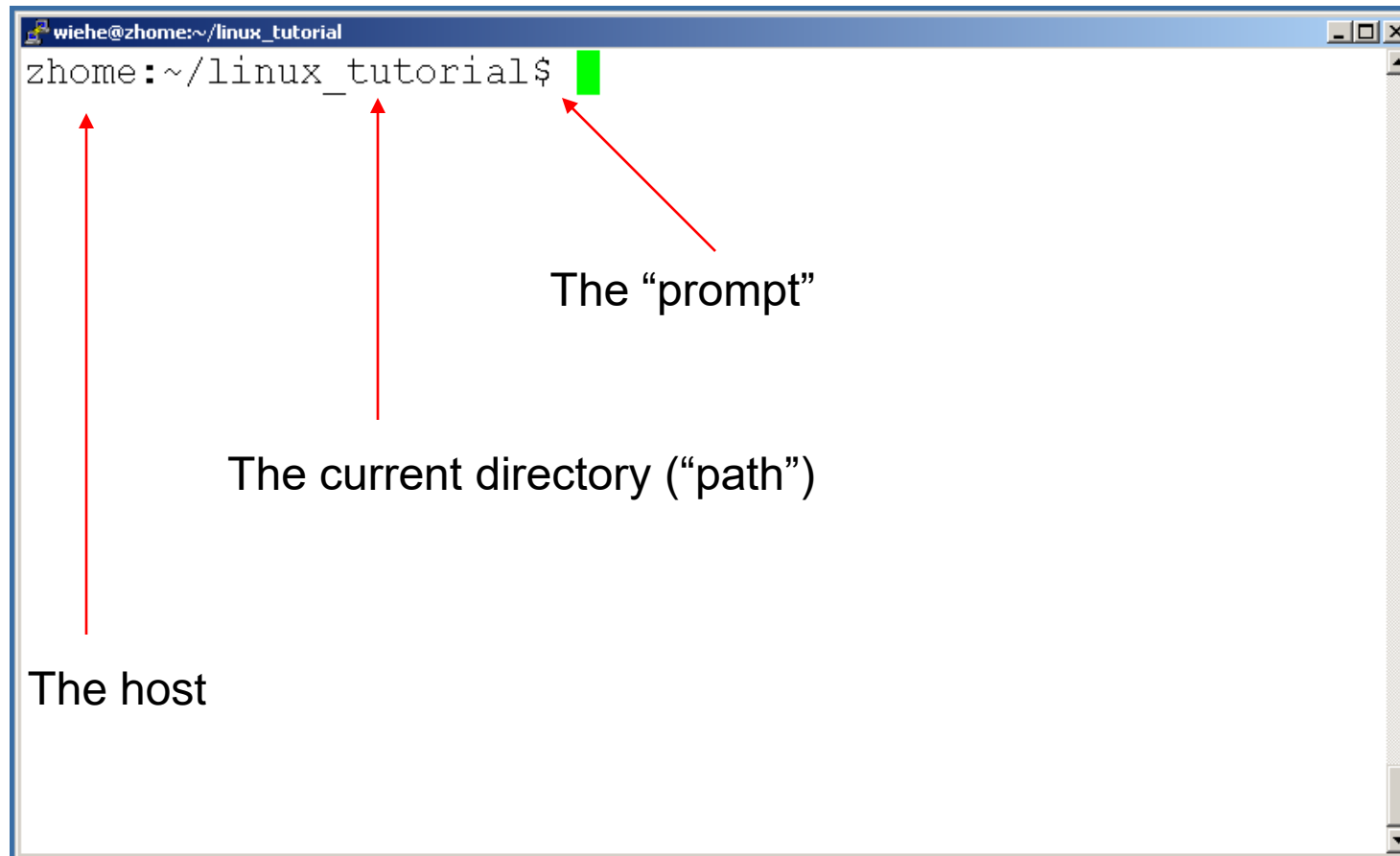
# Connecting to a Unix/Linux system

- Open up a terminal:



# Connecting to a Unix/Linux system

- Open up a terminal:



# What exactly is a “shell”?

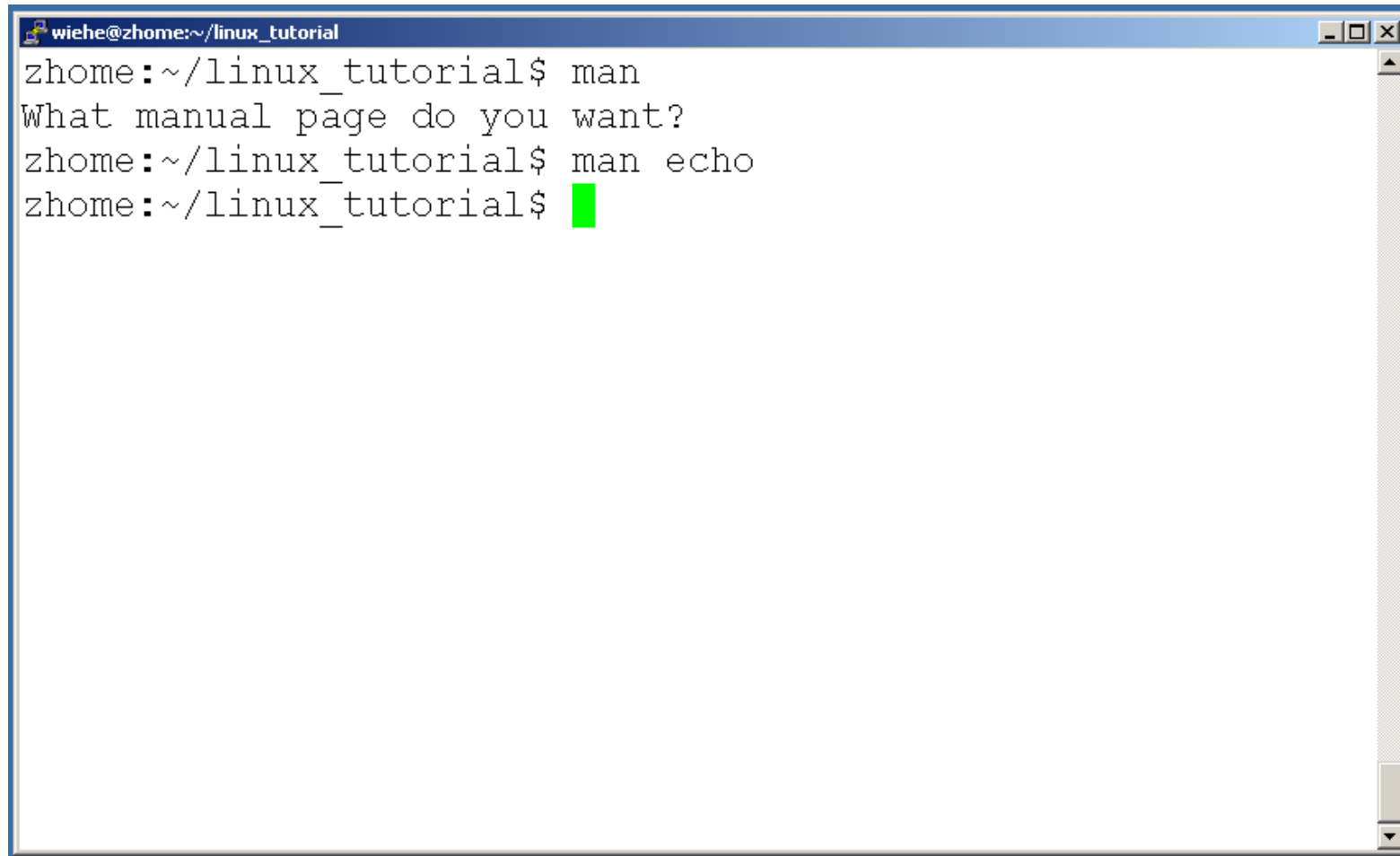
- After logging in, Linux/Unix starts another program called the **shell**
- The shell interprets commands the user types and manages their execution
  - The shell communicates with the internal part of the operating system called the **kernel**
  - The most popular shells are: tcsh, csh, korn, and bash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



# Help!

- Whenever you need help with a command type “man” and the command name

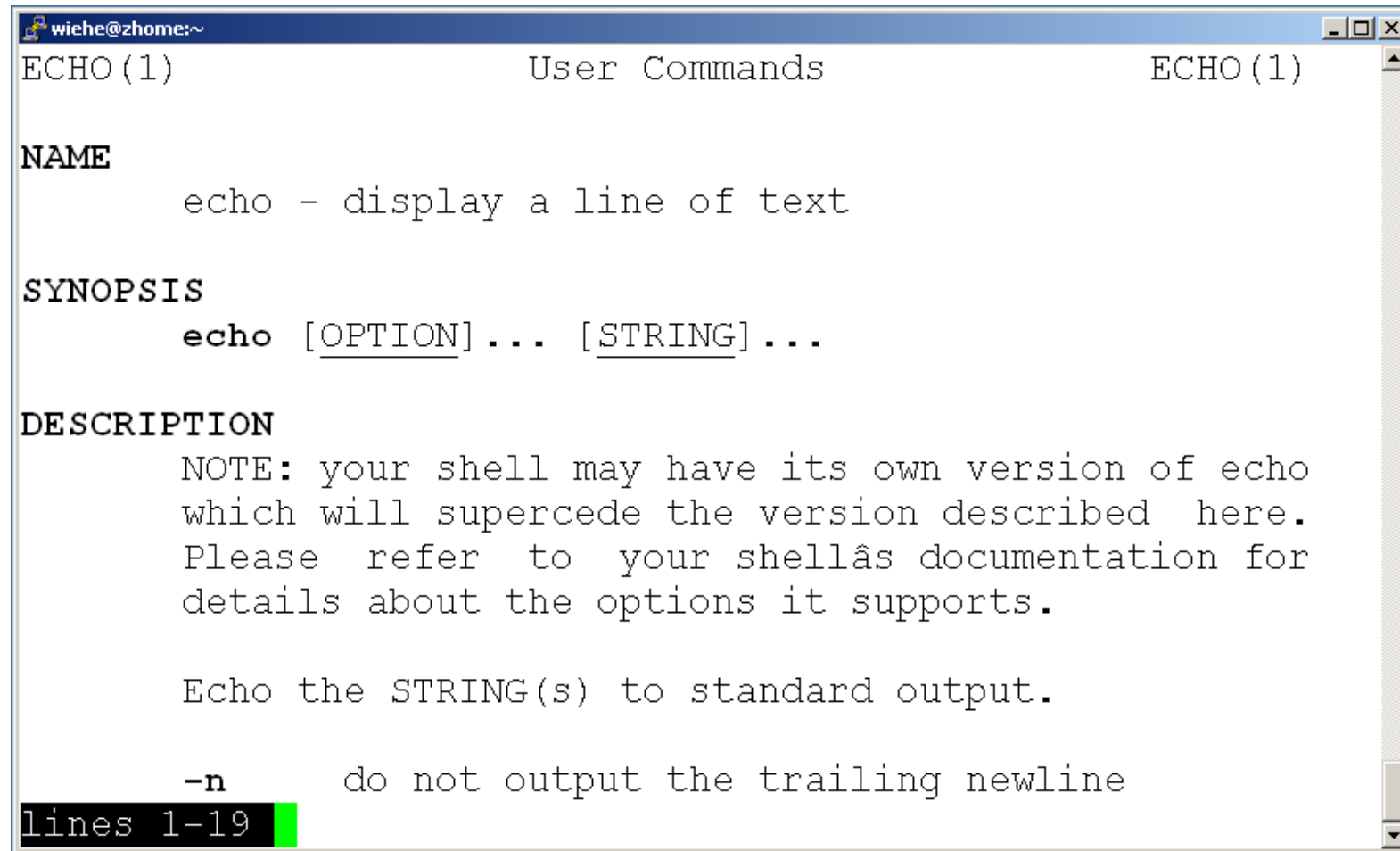
# Help!

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows the following sequence of commands and output:

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

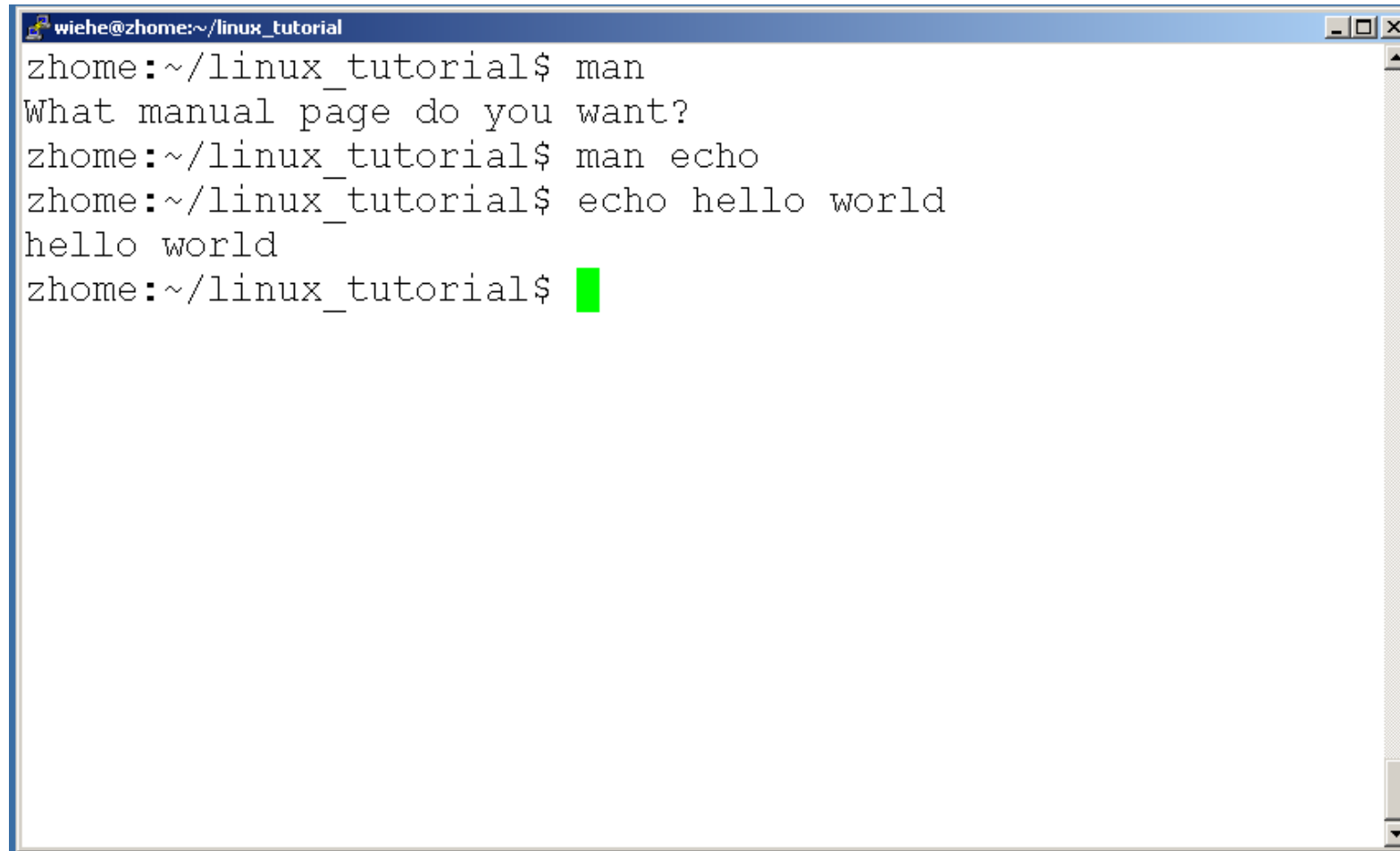
A green cursor is visible at the end of the last line.

# 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 shell's 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
```

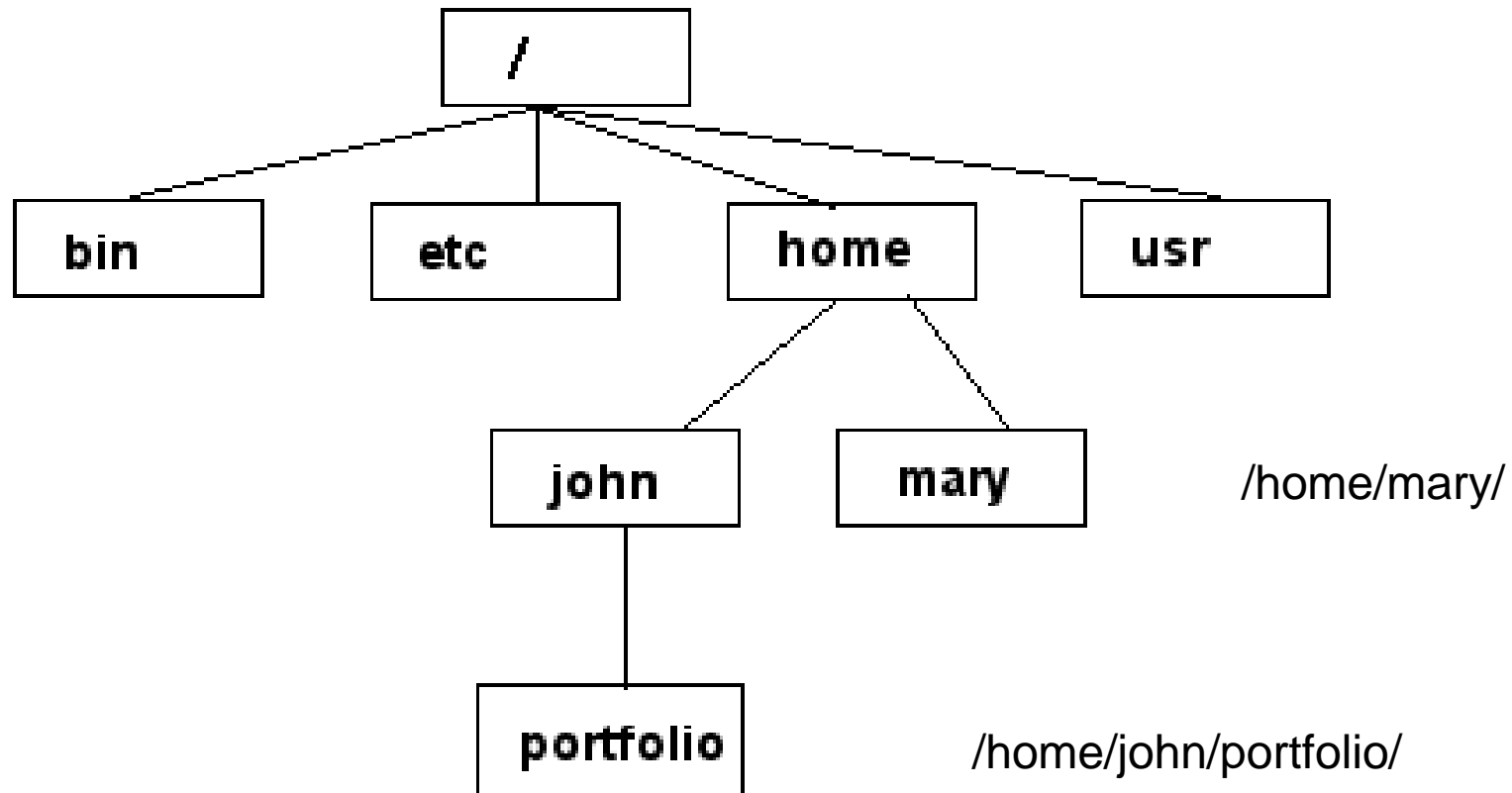
# Help!

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows a sequence of commands and their outputs: 'man' is entered, followed by 'What manual page do you want?'; then 'man echo' is entered; then 'echo hello world' is entered, resulting in the output 'hello world'; finally, the prompt 'zhome:~/linux\_tutorial\$' is shown with a green cursor.

```
wiehe@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$
```

# Unix/Linux File System

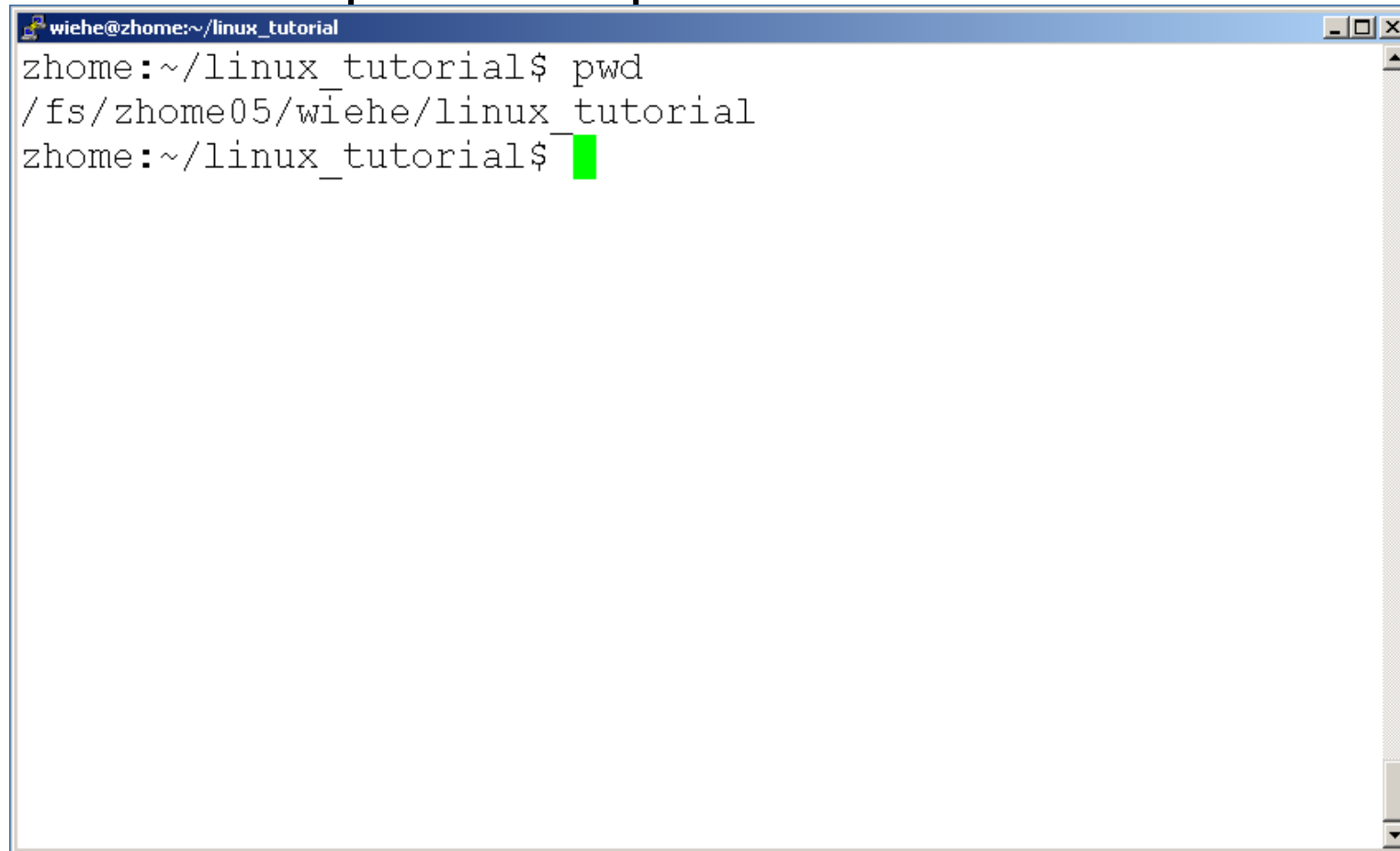
NOTE: Unix file names  
are **CASE SENSITIVE!**



The Path

# Command: pwd

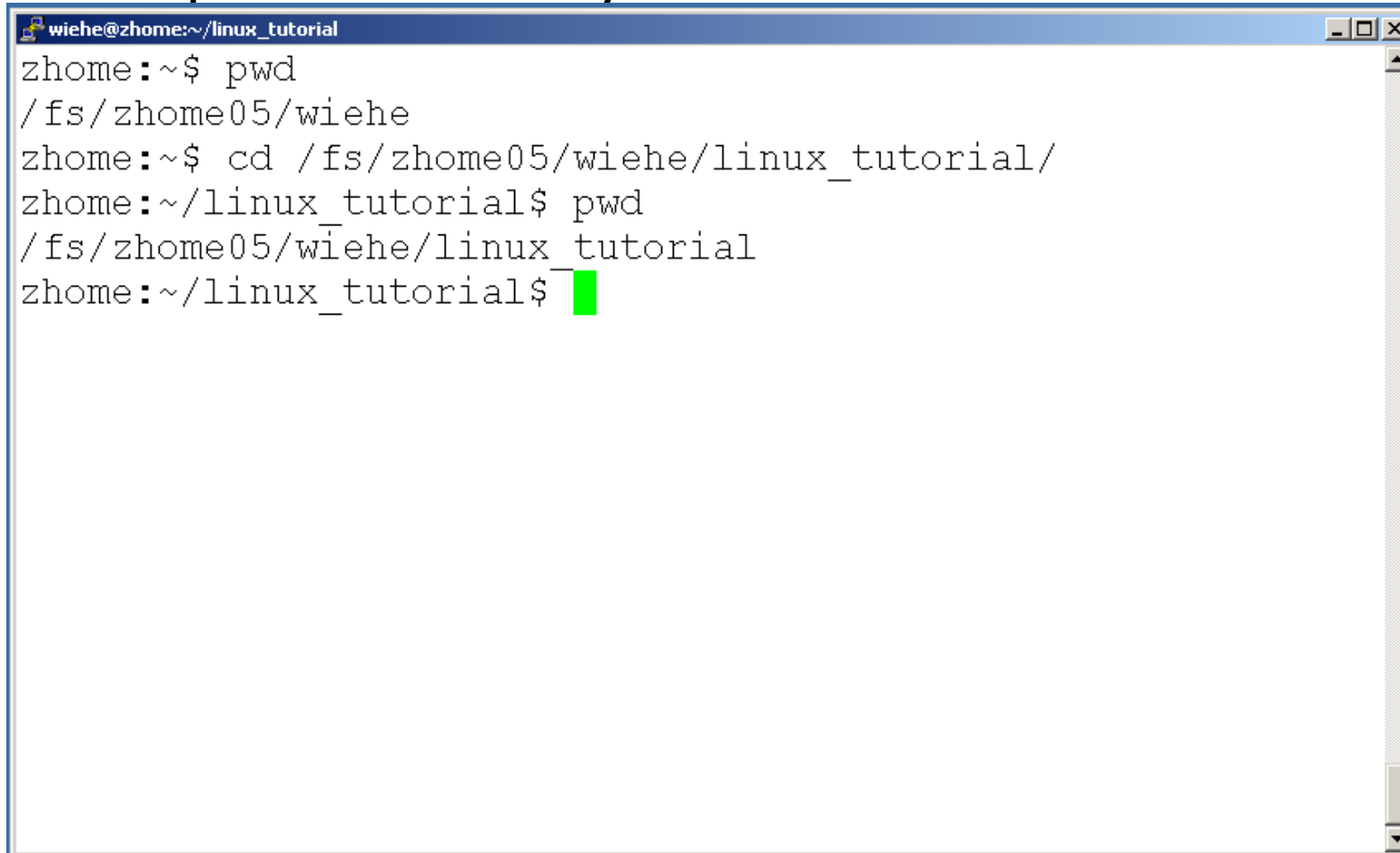
- To find your current path use “pwd”

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows a prompt 'zhome:~/linux\_tutorial\$' followed by the command 'pwd'. The output is '/fs/zhome05/wiehe/linux\_tutorial'. Below the output, the prompt 'zhome:~/linux\_tutorial\$' is shown again with a green cursor. The window has standard Linux window controls (minimize, maximize, close) in the top right corner and a vertical scrollbar on the right side.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ pwd
/fs/zhome05/wiehe/linux_tutorial
zhome:~/linux_tutorial$
```

# Command: cd

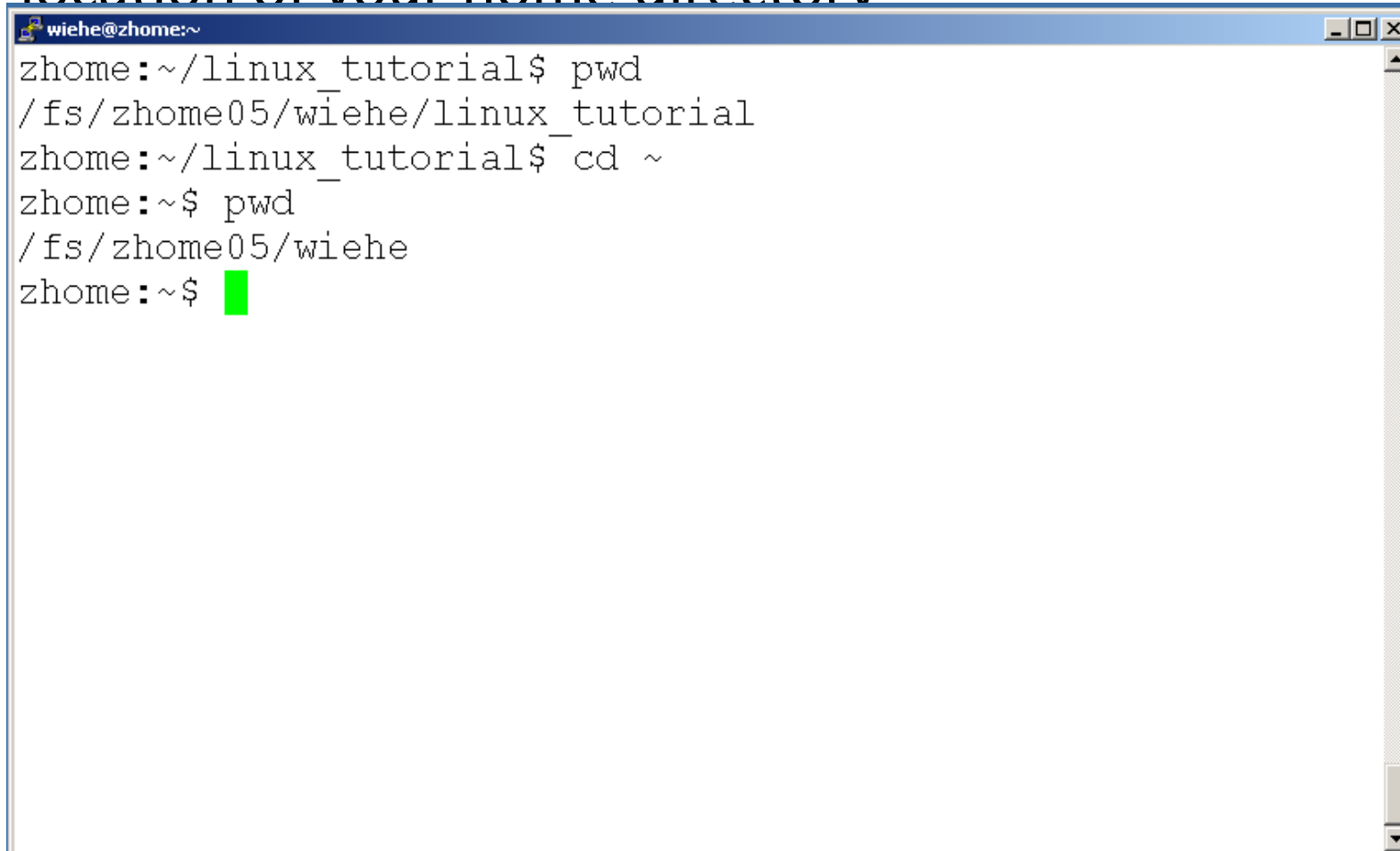
- To change to a specific directory use “cd”

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' with standard window controls. It displays a sequence of commands and their outputs: 'pwd' returns '/fs/zhome05/wiehe', 'cd /fs/zhome05/wiehe/linux\_tutorial/' changes the directory, and a second 'pwd' returns '/fs/zhome05/wiehe/linux\_tutorial'. A green cursor is at the end of the final prompt.

```
wiehe@zhome:~/linux_tutorial
zhome:~$ pwd
/fs/zhome05/wiehe
zhome:~$ cd /fs/zhome05/wiehe/linux_tutorial/
zhome:~/linux_tutorial$ pwd
/fs/zhome05/wiehe/linux_tutorial
zhome:~/linux_tutorial$
```

# Command: cd

- “~” is the location of your home directory

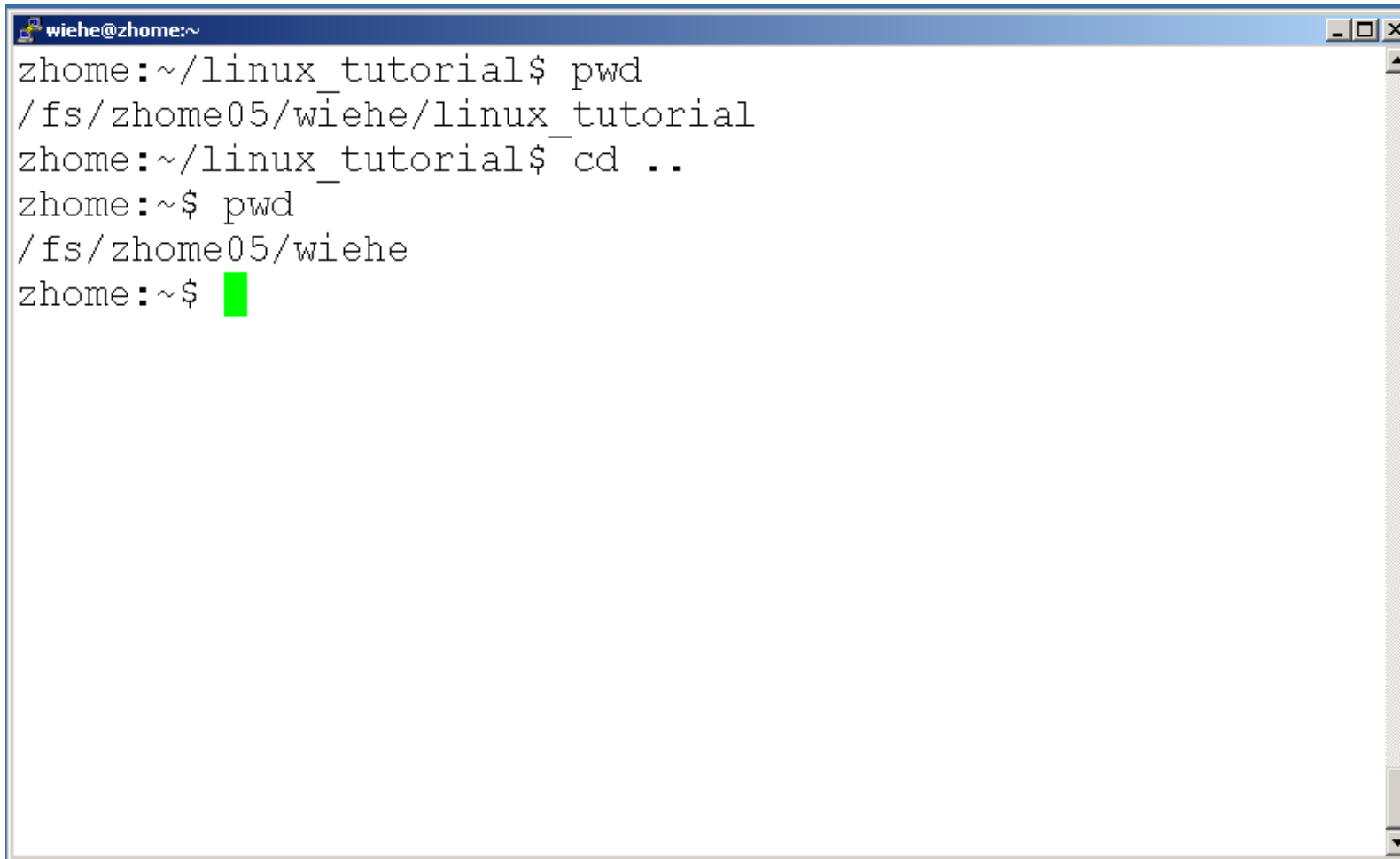


```
wiehe@zhome:~  
zhome:~/linux_tutorial$ pwd  
/fs/zhome05/wiehe/linux_tutorial  
zhome:~/linux_tutorial$ cd ~  
zhome:~$ pwd  
/fs/zhome05/wiehe  
zhome:~$
```



# Command: cd

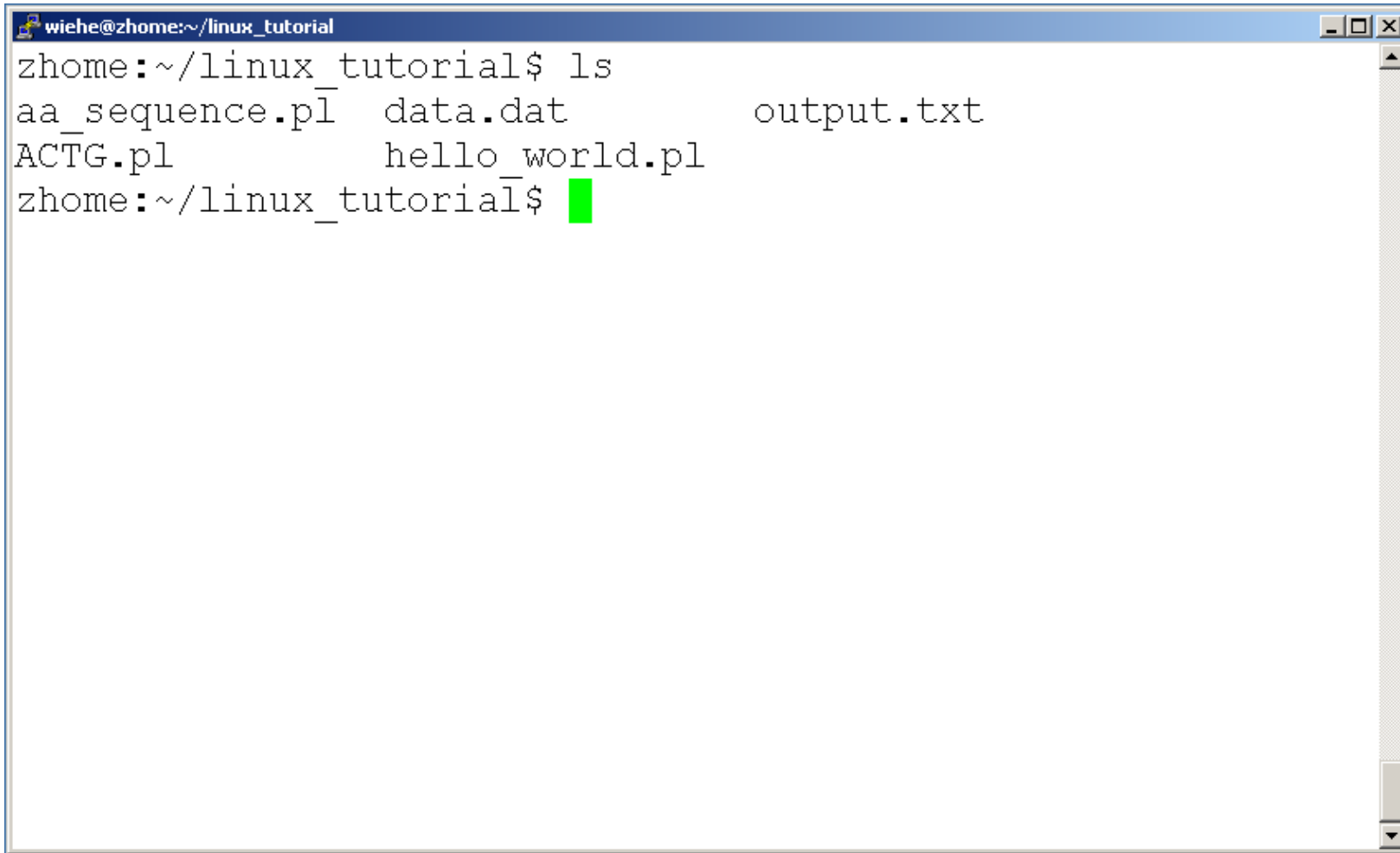
- “..” is the location of the directory below current one

A terminal window titled 'wiehe@zhome:~' showing a sequence of commands and their outputs. The user starts in the directory ~/linux\_tutorial, runs 'pwd' to see the full path, then runs 'cd ..' to move to the parent directory. Finally, they run 'pwd' again to confirm they are now in the home directory. A green cursor is visible at the end of the last command line.

```
wiehe@zhome:~  
zhome:~/linux_tutorial$ pwd  
/fs/zhome05/wiehe/linux_tutorial  
zhome:~/linux_tutorial$ cd ..  
zhome:~$ pwd  
/fs/zhome05/wiehe  
zhome:~$ █
```

# Command: ls

- To list the files in the current directory use “ls”

A screenshot of a Linux terminal window. The title bar at the top reads 'wiehe@zhome:~/linux\_tutorial'. The terminal shows the command 'ls' being executed, which lists the files 'aa\_sequence.pl', 'data.dat', 'output.txt', 'ACTG.pl', and 'hello\_world.pl'. The prompt 'zhome:~/linux\_tutorial\$' is shown at the bottom with a green cursor.

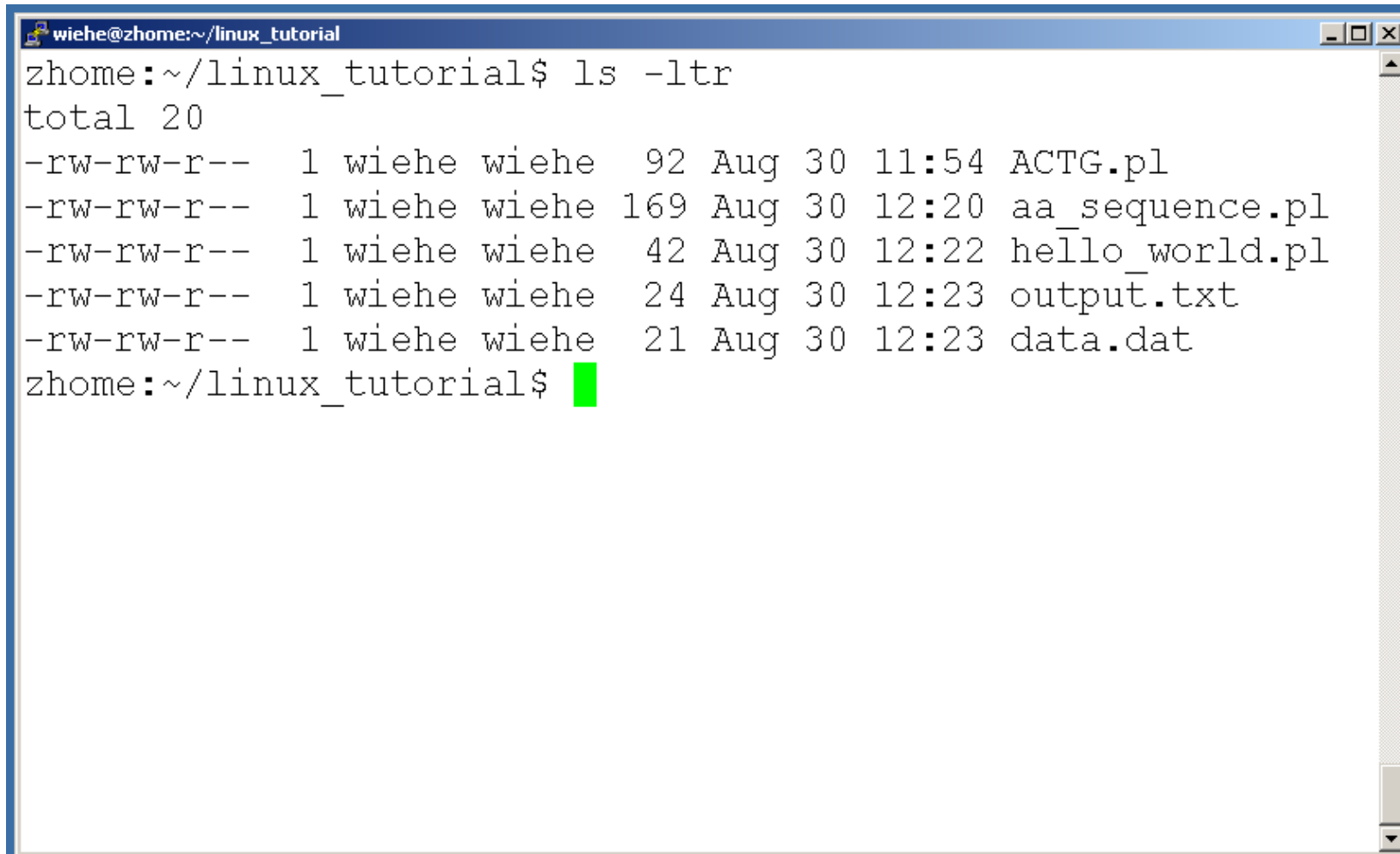
```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat          output.txt
ACTG.pl        hello_world.pl
zhome:~/linux_tutorial$
```

# Command: ls

- ls has many options
  - -l long list (displays lots of info)
  - -t sort by modification time
  - -S sort by size
  - -h list file sizes in human readable format
  - -r reverse the order
- “man ls” for more options
- Options can be combined: “ls -ltr”

# Command: ls -ltr

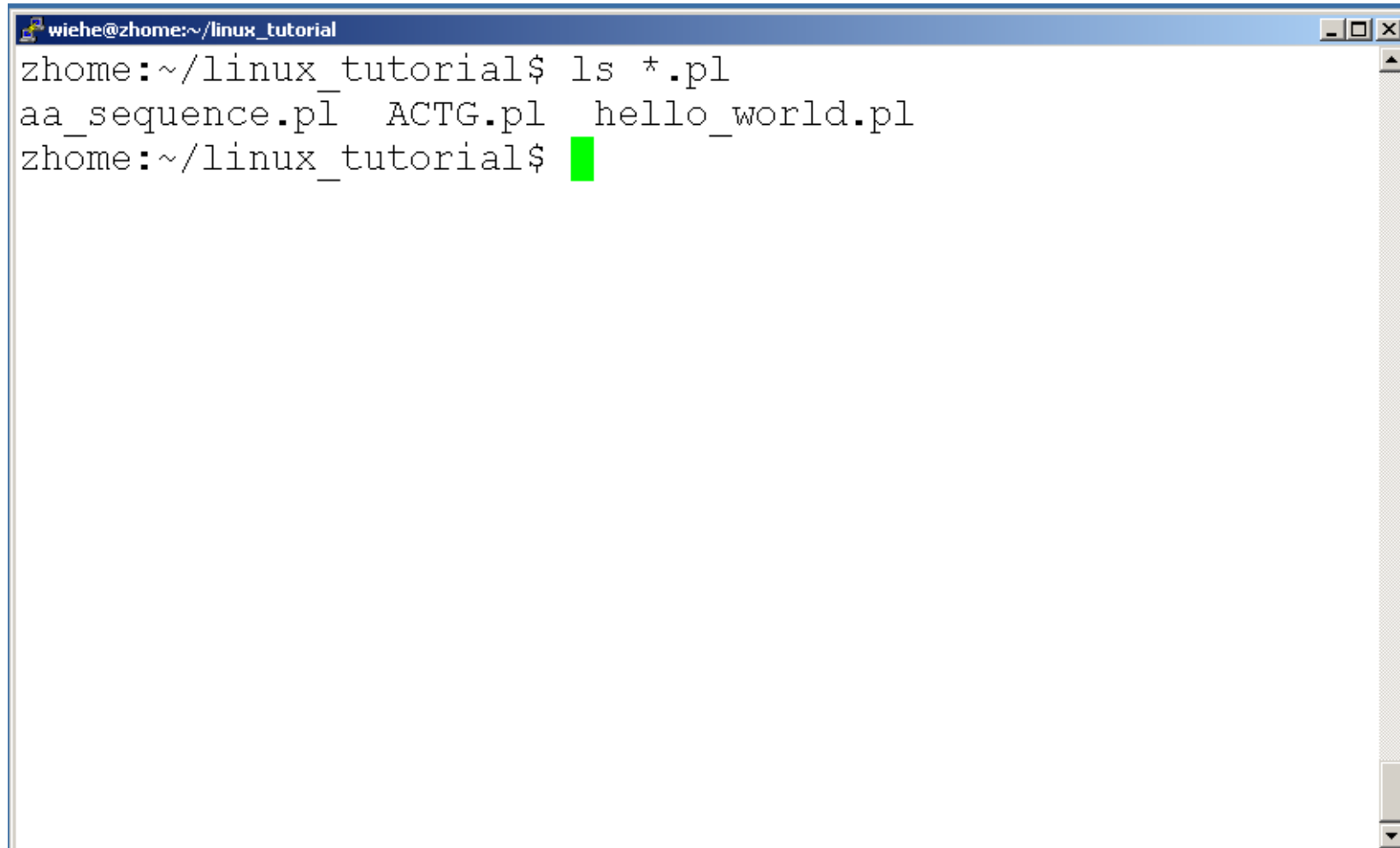
- List files by time in reverse order with long listing

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing the output of the 'ls -ltr' command. The output lists five files in reverse chronological order: ACTG.pl, aa\_sequence.pl, hello\_world.pl, output.txt, and data.dat. Each line shows permissions, file size, owner, group, date, time, and filename. The prompt 'zhome:~/linux\_tutorial\$' is followed by a green cursor.

```
wiehe@zhome:~/linux_tutorial$ ls -ltr
total 20
-rw-rw-r-- 1 wiehe wiehe  92 Aug 30 11:54 ACTG.pl
-rw-rw-r-- 1 wiehe wiehe 169 Aug 30 12:20 aa_sequence.pl
-rw-rw-r-- 1 wiehe wiehe  42 Aug 30 12:22 hello_world.pl
-rw-rw-r-- 1 wiehe wiehe  24 Aug 30 12:23 output.txt
-rw-rw-r-- 1 wiehe wiehe  21 Aug 30 12:23 data.dat
zhome:~/linux_tutorial$
```

# General Syntax: \*

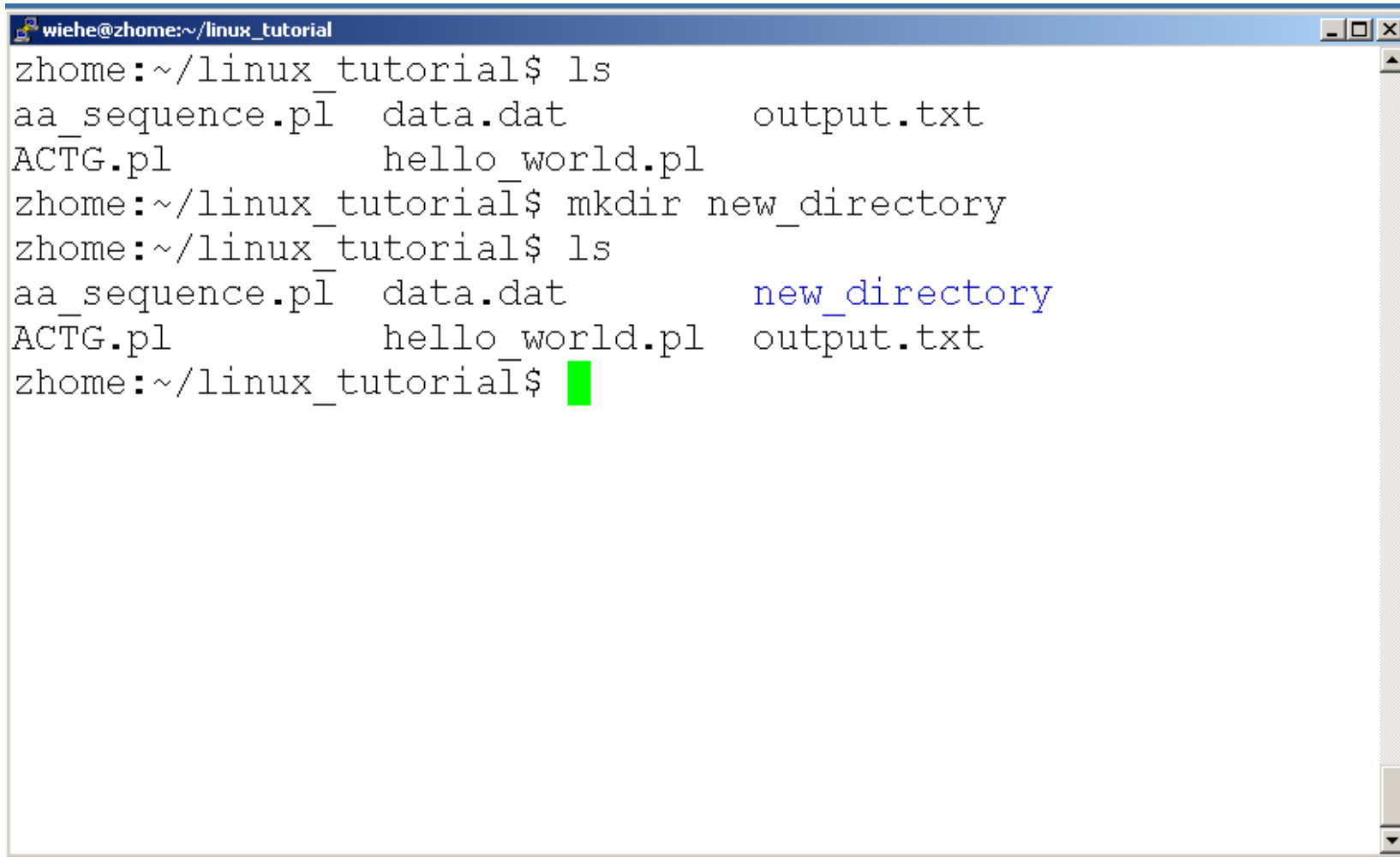
- “\*” can be used as a wildcard in unix/linux

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing a command prompt. The user enters 'ls \*.pl' and the terminal outputs 'aa\_sequence.pl ACTG.pl hello\_world.pl'. The prompt then returns to 'zhome:~/linux\_tutorial\$' with a green cursor.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls *.pl
aa_sequence.pl  ACTG.pl  hello_world.pl
zhome:~/linux_tutorial$
```

# Command: mkdir

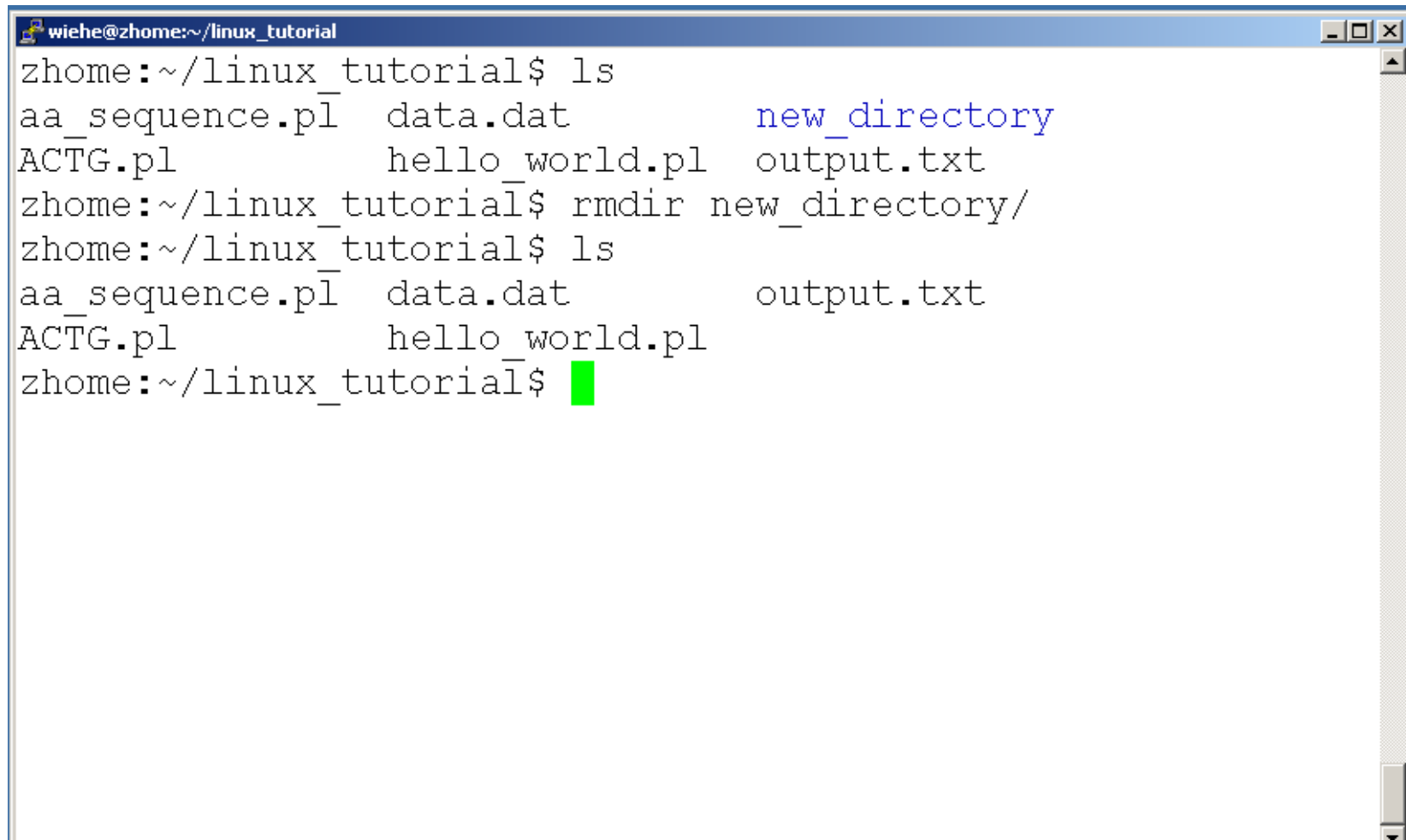
- To create a new directory use “mkdir”

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing the execution of the 'mkdir' command. The window has a blue title bar and standard window controls. The terminal text shows the initial directory listing, the command to create 'new\_directory', and the subsequent directory listing which now includes 'new\_directory' in blue text. A green cursor is visible at the end of the last command line.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat      output.txt
ACTG.pl        hello_world.pl
zhome:~/linux_tutorial$ mkdir new_directory
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat      new_directory
ACTG.pl        hello_world.pl output.txt
zhome:~/linux_tutorial$ █
```

# Command: rmdir

- To remove an empty directory use “rmdir”



```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat          new_directory
ACTG.pl         hello_world.pl    output.txt
zhome:~/linux_tutorial$ rmdir new_directory/
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat          output.txt
ACTG.pl         hello_world.pl
zhome:~/linux_tutorial$
```

# Displaying a file

- Various ways to display a file in Unix
  - cat
  - less
  - head
  - tail



# Command: cat

- Dumps an entire file to standard output
- Good for displaying short, simple files

# Command: less

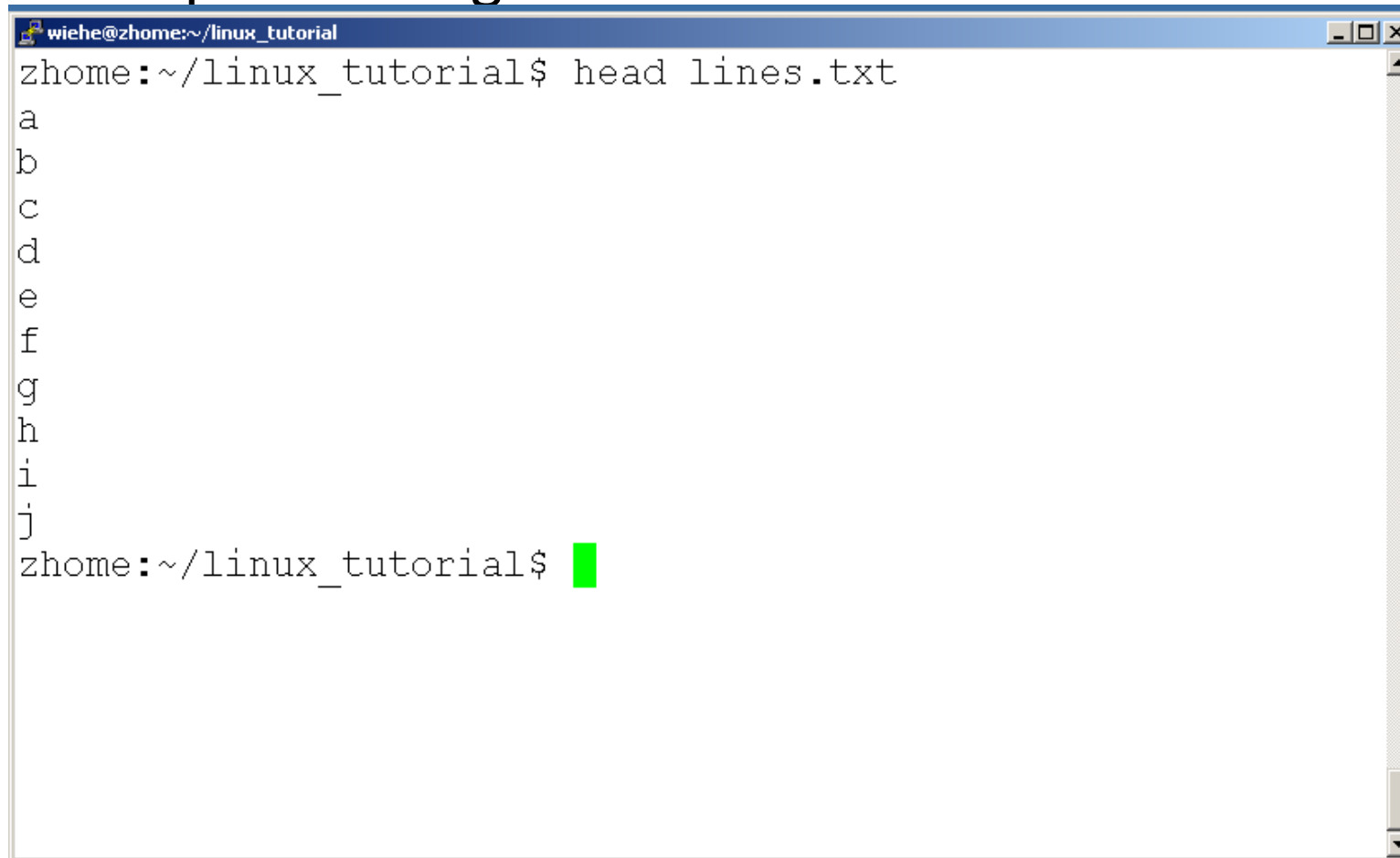
- “less” displays a file, allowing forward/backward movement within it
  - return scrolls forward one line, space one page
  - y scrolls back one line, b one page
- use “/” to search for a string
- Press q to quit

# Command: head

- “head” displays the top part of a file
- By default it shows the first 10 lines
- -n option allows you to change that
- “head -n50 file.txt” displays the first 50 lines of file.txt

# Command: head

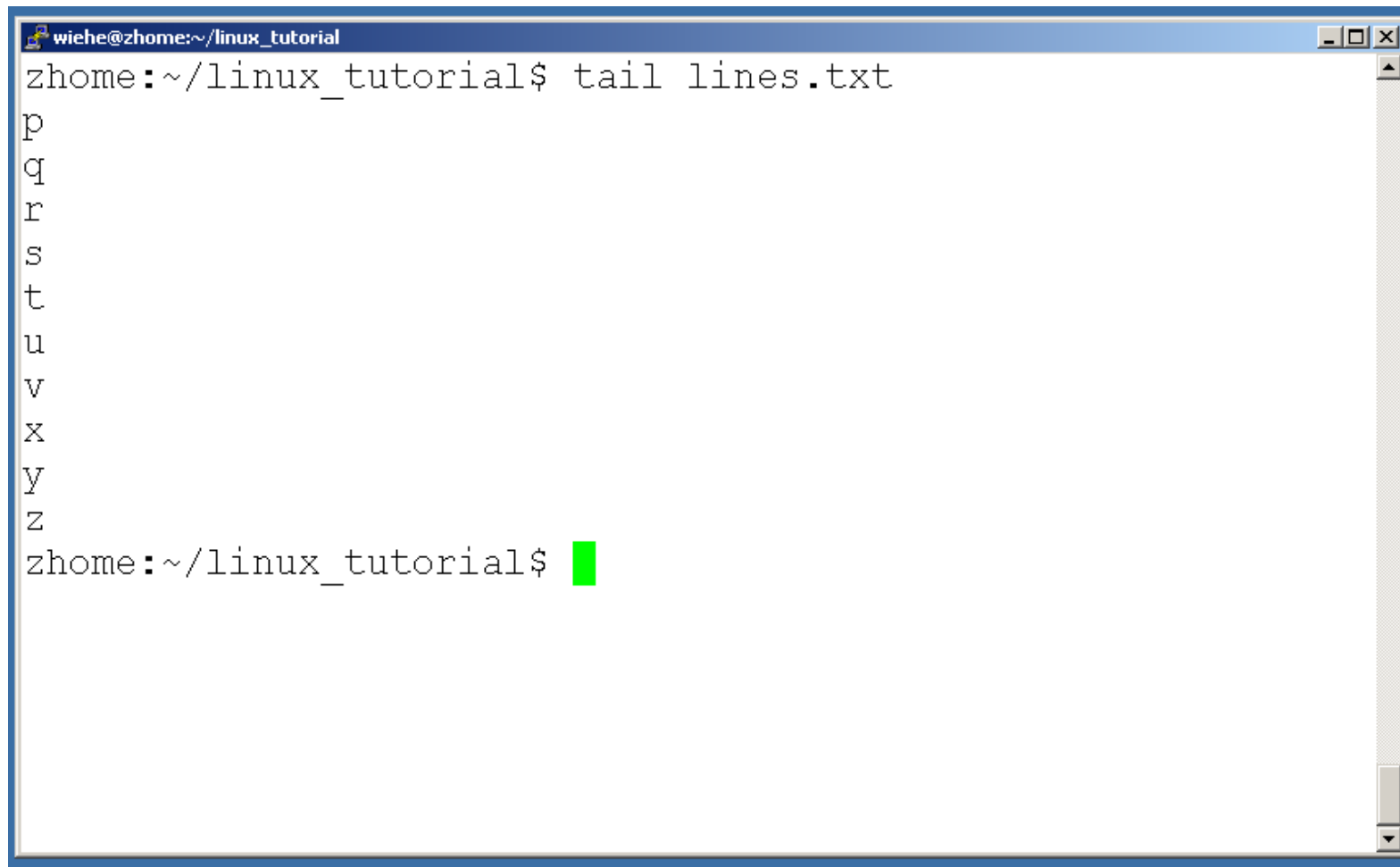
- Here's an example of using "head":

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows the command 'head lines.txt' being executed, which outputs the first ten lines of a file: 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', and 'j'. The prompt 'zhome:~/linux\_tutorial\$' is shown again with a green cursor.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ head lines.txt
a
b
c
d
e
f
g
h
i
j
zhome:~/linux_tutorial$
```

# Command: tail

- Same as head, but shows the last lines

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows the command 'tail lines.txt' being executed. The output consists of the letters 'p', 'q', 'r', 's', 't', 'u', 'v', 'x', 'y', and 'z' on separate lines. The prompt 'zhome:~/linux\_tutorial\$' is followed by a green cursor block.

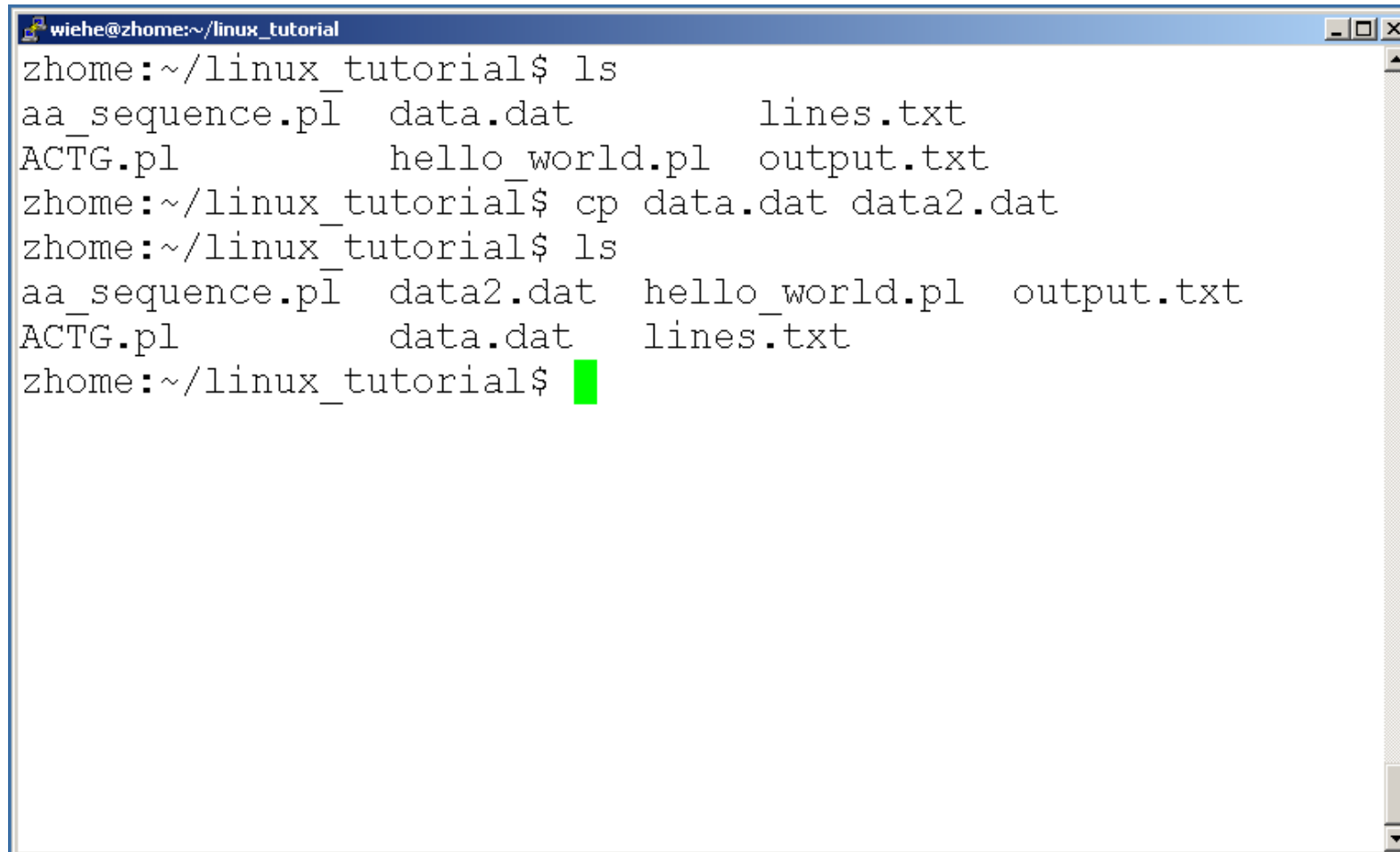
```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ tail lines.txt
p
q
r
s
t
u
v
x
y
z
zhome:~/linux_tutorial$
```

# File Commands

- Copying a file: cp
- Move or rename a file: mv
- Remove a file: rm

# Command: cp

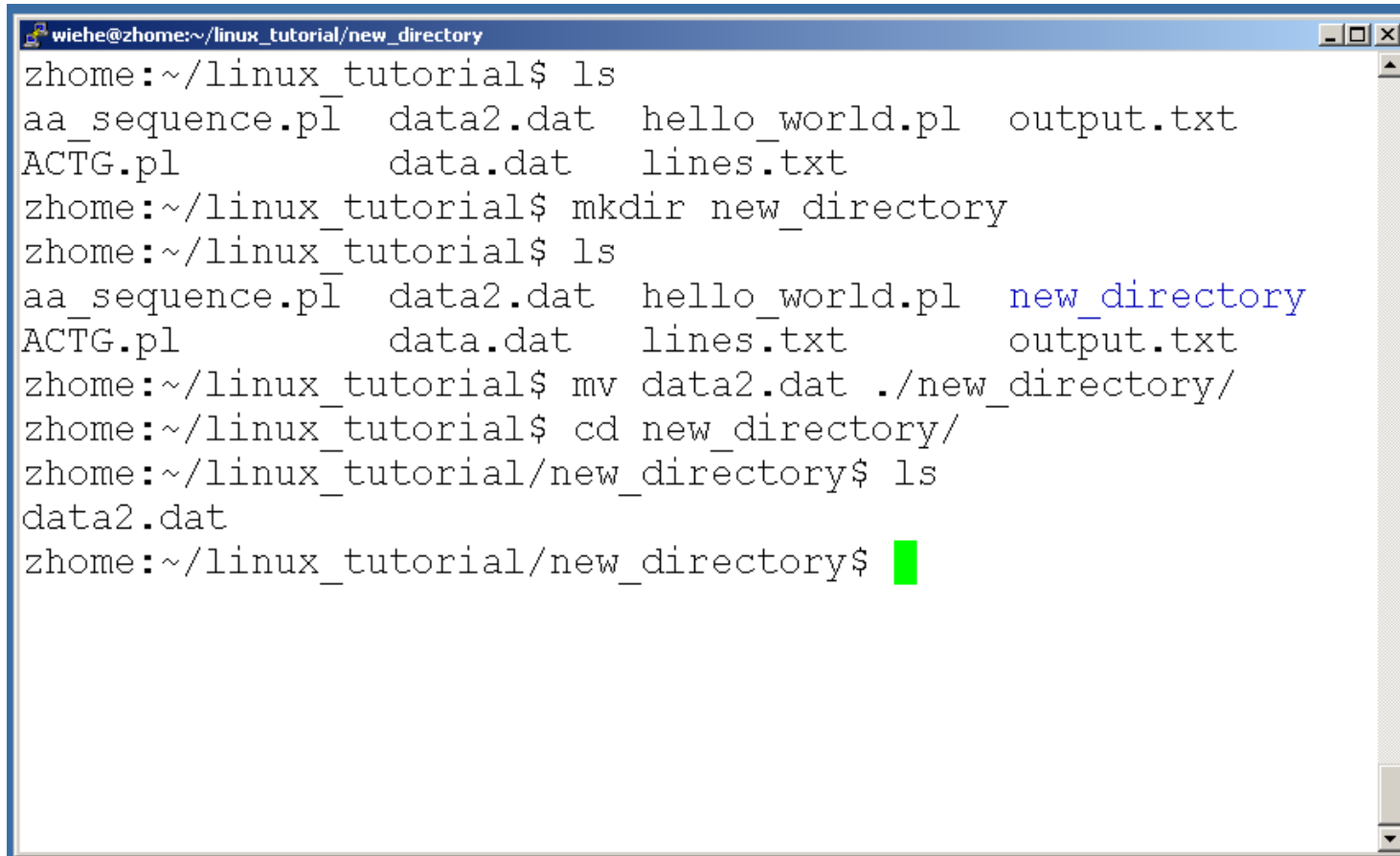
- To copy a file use “cp”



```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat      lines.txt
ACTG.pl        hello_world.pl output.txt
zhome:~/linux_tutorial$ cp data.dat data2.dat
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data2.dat    hello_world.pl  output.txt
ACTG.pl        data.dat     lines.txt
zhome:~/linux_tutorial$
```

# Command: mv

- To move a file to a different location use “mv”

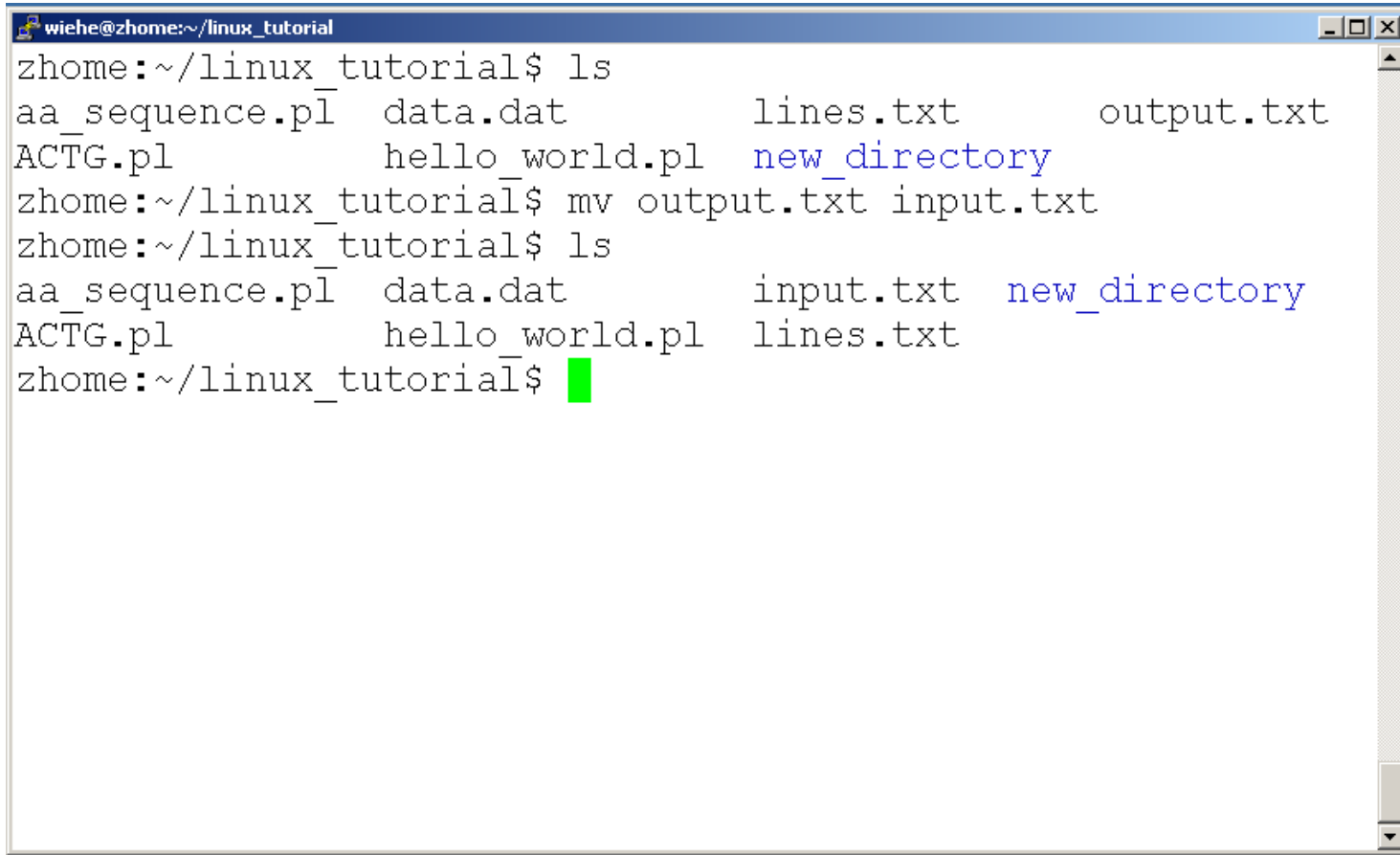
A terminal window titled 'wiehe@zhome:~/linux\_tutorial/new\_directory' showing a series of commands and their outputs. The commands include 'ls', 'mkdir new\_directory', 'ls', 'mv data2.dat ./new\_directory/', 'cd new\_directory/', and another 'ls'. The output shows the creation of the 'new\_directory' and the successful movement of 'data2.dat' into it.

```
wiehe@zhome:~/linux_tutorial/new_directory
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data2.dat  hello_world.pl  output.txt
ACTG.pl        data.dat   lines.txt
zhome:~/linux_tutorial$ mkdir new_directory
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data2.dat  hello_world.pl  new_directory
ACTG.pl        data.dat   lines.txt       output.txt
zhome:~/linux_tutorial$ mv data2.dat ./new_directory/
zhome:~/linux_tutorial$ cd new_directory/
zhome:~/linux_tutorial/new_directory$ ls
data2.dat
zhome:~/linux_tutorial/new_directory$
```



# Command: mv

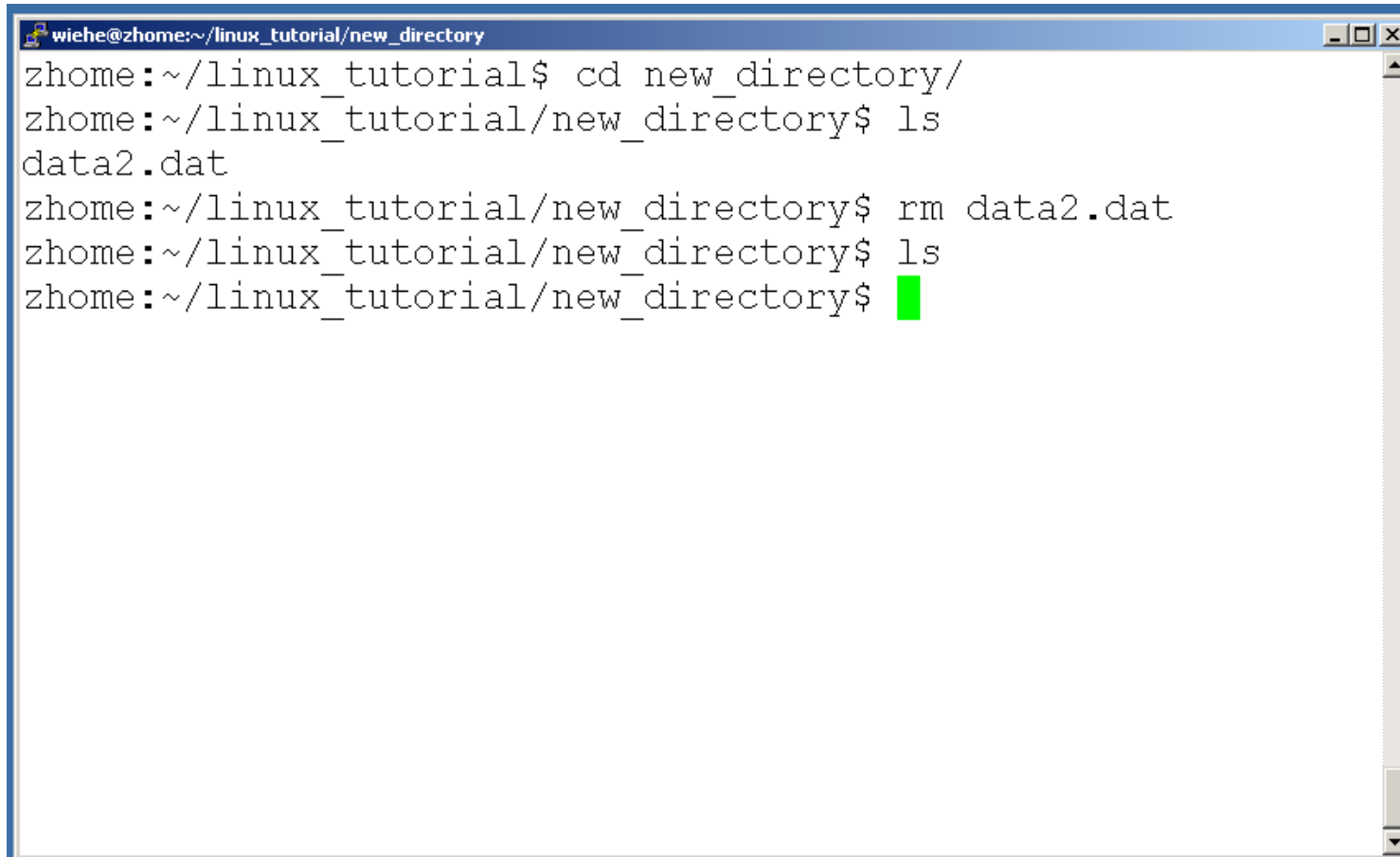
- mv can also be used to rename a file

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing a sequence of commands and their outputs. The user runs 'ls' to list files, then 'mv output.txt input.txt' to rename 'output.txt' to 'input.txt', and finally 'ls' again to confirm the change. The files listed are 'aa\_sequence.pl', 'data.dat', 'lines.txt', 'output.txt' (before the move), 'ACTG.pl', 'hello\_world.pl', and 'new\_directory'.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat      lines.txt      output.txt
ACTG.pl        hello_world.pl new_directory
zhome:~/linux_tutorial$ mv output.txt input.txt
zhome:~/linux_tutorial$ ls
aa_sequence.pl  data.dat      input.txt      new_directory
ACTG.pl        hello_world.pl lines.txt
zhome:~/linux_tutorial$
```

# Command: rm

- To remove a file use “rm”



```
wiehe@zhome:~/linux_tutorial/new_directory
zhome:~/linux_tutorial$ cd new_directory/
zhome:~/linux_tutorial/new_directory$ ls
data2.dat
zhome:~/linux_tutorial/new_directory$ rm data2.dat
zhome:~/linux_tutorial/new_directory$ ls
zhome:~/linux_tutorial/new_directory$
```

# Command: rm

- To remove a file “recursively”: `rm -r`
- Used to remove all files and directories
- Be very careful, deletions are permanent in Unix/Linux

# 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

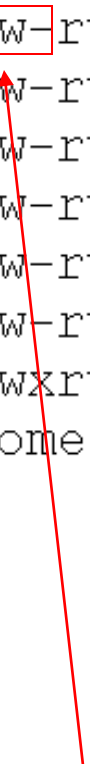
# Permission levels

- “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

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls -l
total 28
-rw-rw-r-- 1 wiehe wiehe 169 Aug 30 12:20 aa_sequence.pl
-rw-rw-r-- 1 wiehe wiehe 92 Aug 30 11:54 ACTG.pl
-rw-rw-r-- 1 wiehe wiehe 21 Aug 30 12:23 data.dat
-rw-rw-r-- 1 wiehe wiehe 42 Aug 30 12:22 hello_world.pl
-rw-rw-r-- 1 wiehe wiehe 24 Aug 30 12:23 input.txt
-rw-rw-r-- 1 wiehe wiehe 50 Aug 30 13:13 lines.txt
drwxrwxr-x 2 wiehe wiehe 4096 Aug 30 13:19 new_directory
zhome:~/linux_tutorial$
```

**User (you)**



# File Permissions

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls -l
total 28
-rw-rw-r-- 1 wiehe wiehe 169 Aug 30 12:20 aa_sequence.pl
-rw-rw-r-- 1 wiehe wiehe 92 Aug 30 11:54 ACTG.pl
-rw-rw-r-- 1 wiehe wiehe 21 Aug 30 12:23 data.dat
-rw-rw-r-- 1 wiehe wiehe 42 Aug 30 12:22 hello_world.pl
-rw-rw-r-- 1 wiehe wiehe 24 Aug 30 12:23 input.txt
-rw-rw-r-- 1 wiehe wiehe 50 Aug 30 13:13 lines.txt
drwxrwxr-x 2 wiehe wiehe 4096 Aug 30 13:19 new_directory
zhome:~/linux_tutorial$
```

**Group**

# File Permissions

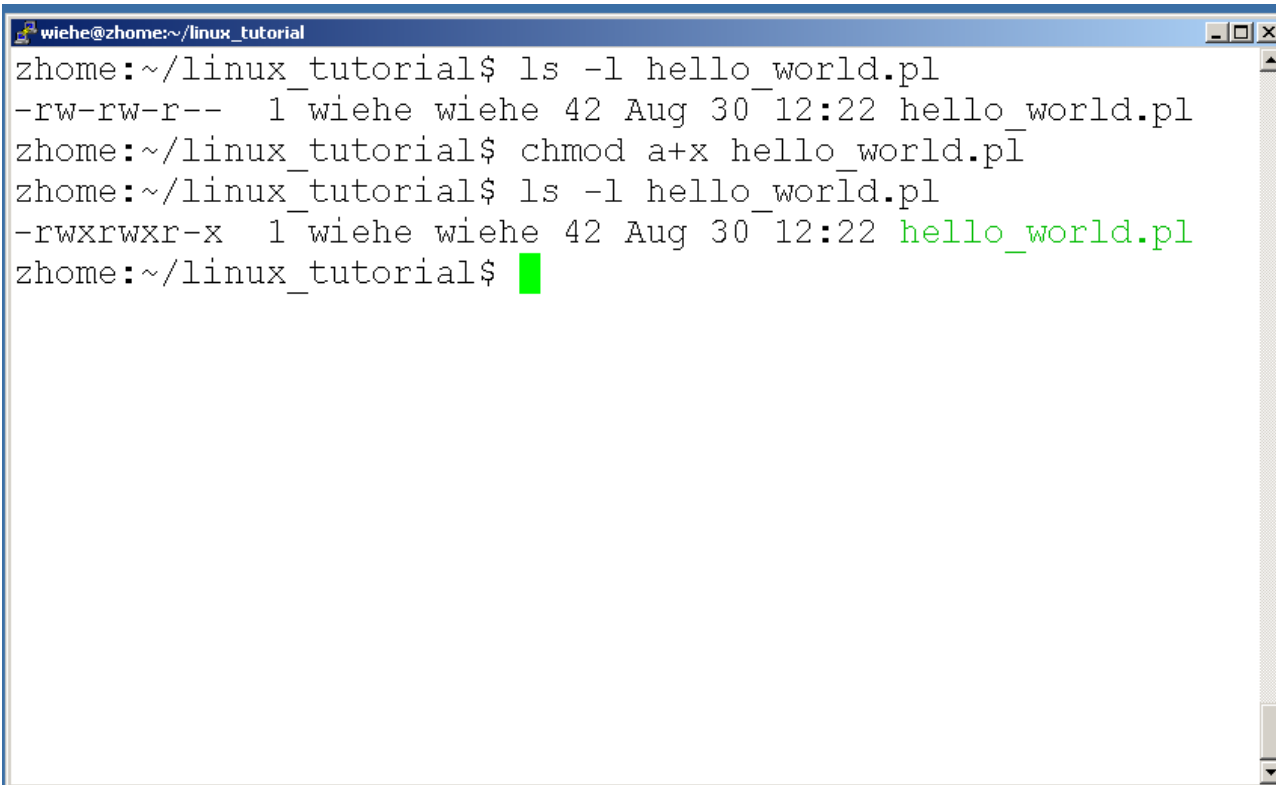
```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls -l
total 28
-rw-rw-r-- 1 wiehe wiehe 169 Aug 30 12:20 aa_sequence.pl
-rw-rw-r-- 1 wiehe wiehe 92 Aug 30 11:54 ACTG.pl
-rw-rw-r-- 1 wiehe wiehe 21 Aug 30 12:23 data.dat
-rw-rw-r-- 1 wiehe wiehe 42 Aug 30 12:22 hello_world.pl
-rw-rw-r-- 1 wiehe wiehe 24 Aug 30 12:23 input.txt
-rw-rw-r-- 1 wiehe wiehe 50 Aug 30 13:13 lines.txt
drwxrwxr-x 2 wiehe wiehe 4096 Aug 30 13:19 new_directory
zhome:~/linux_tutorial$
```

**“The World”**



# Command: chmod

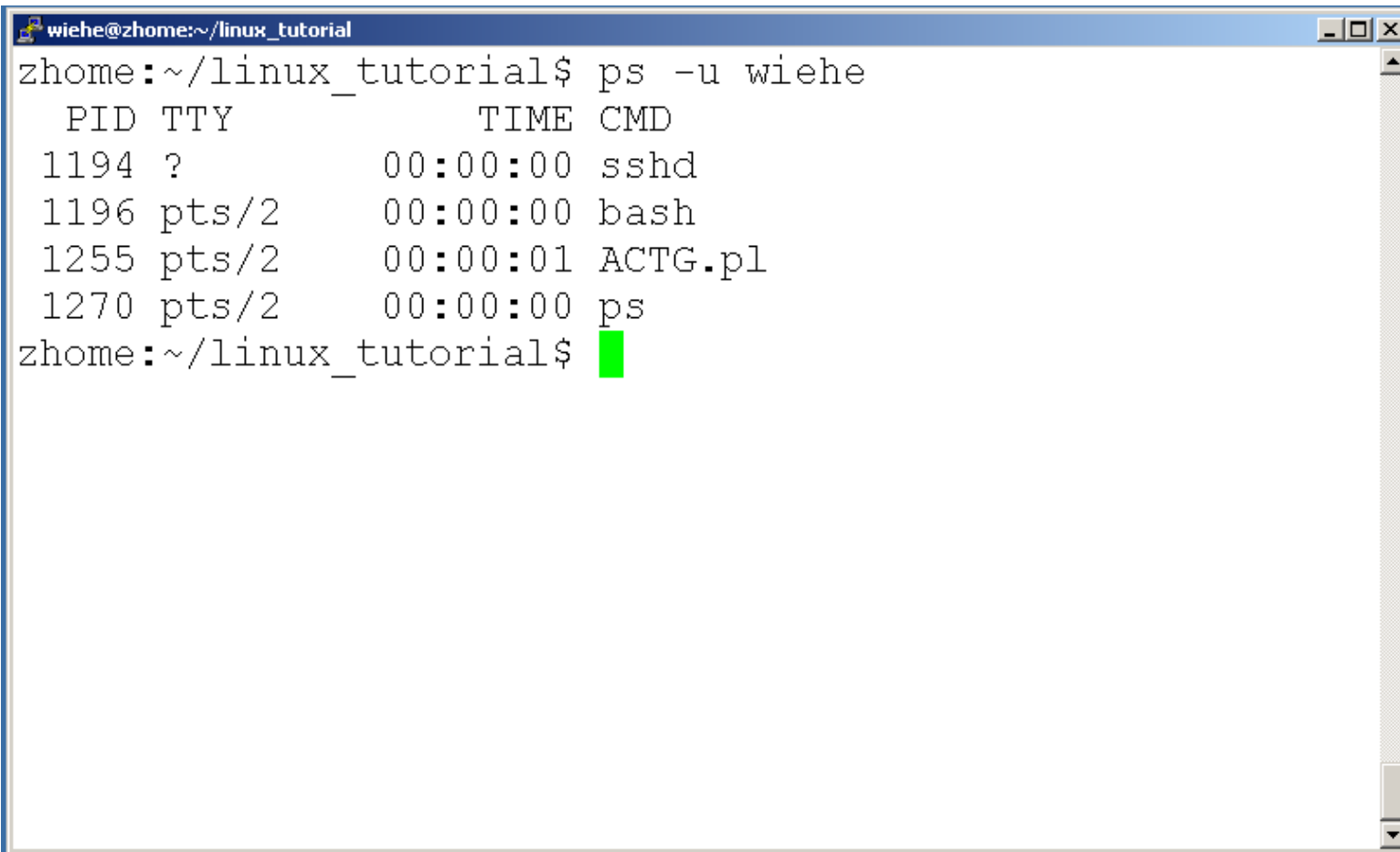
- If you own the file, you can change it's permissions with “chmod”
  - Syntax: `chmod [user/group/others/all]+[permission] [file(s)]`
  - Below we grant execute permission to all:

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing the execution of the 'chmod' command. The user first runs 'ls -l hello\_world.pl' showing permissions '-rw-rw-r--'. Then they run 'chmod a+x hello\_world.pl'. Finally, they run 'ls -l hello\_world.pl' again, showing the updated permissions '-rwxrwxr-x' and the filename 'hello\_world.pl' in green. A green cursor is visible at the end of the last command line.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls -l hello_world.pl
-rw-rw-r-- 1 wiehe wiehe 42 Aug 30 12:22 hello_world.pl
zhome:~/linux_tutorial$ chmod a+x hello_world.pl
zhome:~/linux_tutorial$ ls -l hello_world.pl
-rwxrwxr-x 1 wiehe wiehe 42 Aug 30 12:22 hello_world.pl
zhome:~/linux_tutorial$
```

# Command: ps

- To view the processes that you're running:



```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ps -u wiehe
  PID TTY          TIME CMD
 1194 ?            00:00:00 sshd
 1196 pts/2        00:00:00 bash
 1255 pts/2        00:00:01 ACTG.pl
 1270 pts/2        00:00:00 ps
zhome:~/linux_tutorial$
```

# Command: top

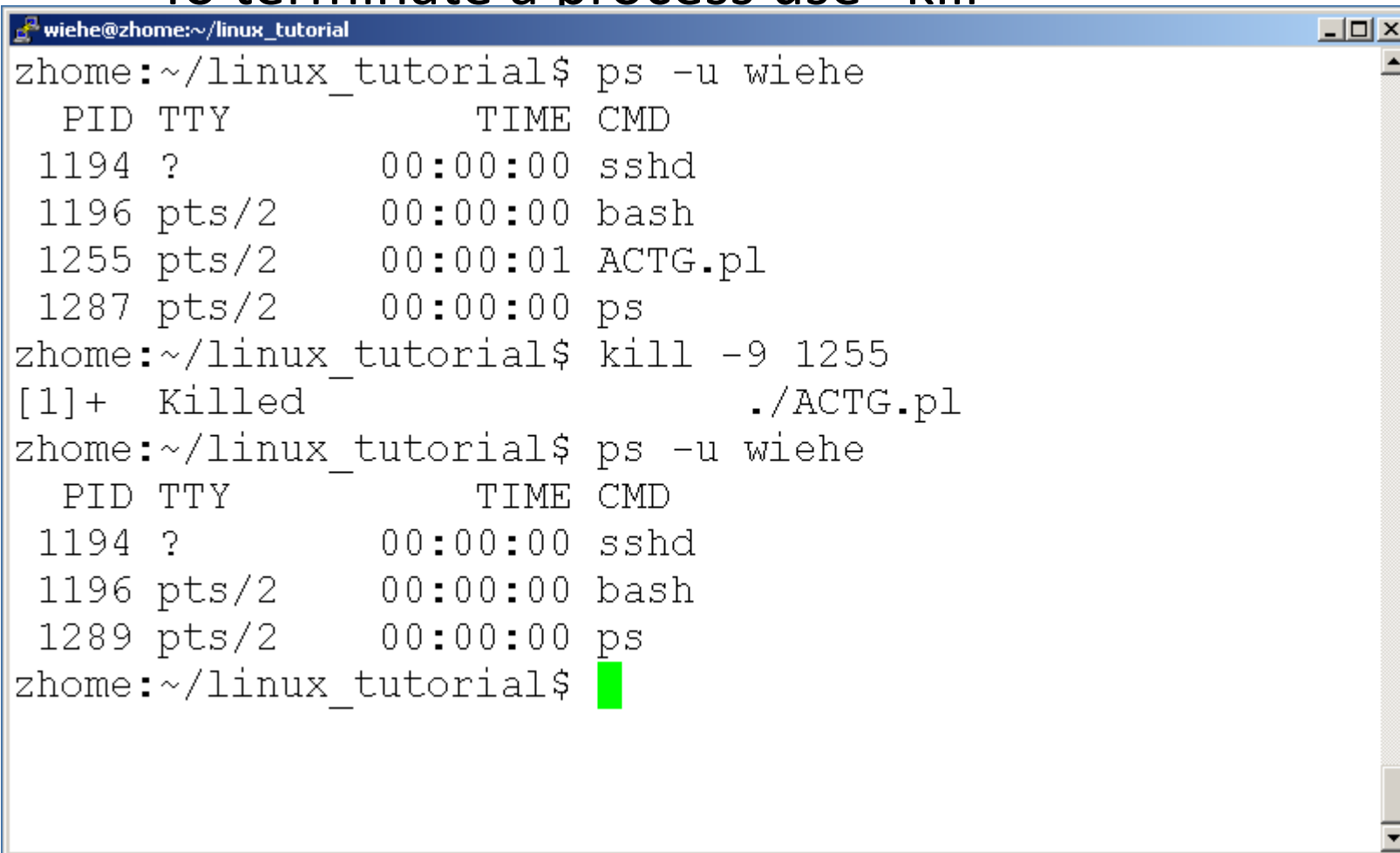
- To view the CPU usage of all processes:

```
wiehe@zhome:~/linux_tutorial
top - 13:46:33 up 50 days,  4:26,  2 users,  load avera
Tasks:  total,      running,      sleeping,      stoppe
Cpu(s):    us,      sy,      ni,      id,      w
Mem:      total,      used,      free,
Swap:      total,      used,      free,

PID USER      PR  NI  VIRT  RES  SHR  S  %CPU  %MEM
3403 root        15   0     0    0    0   S   0.7   0.0
  1 root        16   0  1604  324  292   S   0.0   0.0
  2 root         RT   0     0    0    0   S   0.0   0.0
  3 root        34  19     0    0    0   S   0.0   0.0
  4 root         RT   0     0    0    0   S   0.0   0.0
  5 root        34  19     0    0    0   S   0.0   0.0
  6 root         RT   0     0    0    0   S   0.0   0.0
  7 root        34  19     0    0    0   S   0.0   0.0
  8 root         RT   0     0    0    0   S   0.0   0.0
  9 root        34  19     0    0    0   S   0.0   0.0
```

# Command: kill

- To terminate a process use “kill”

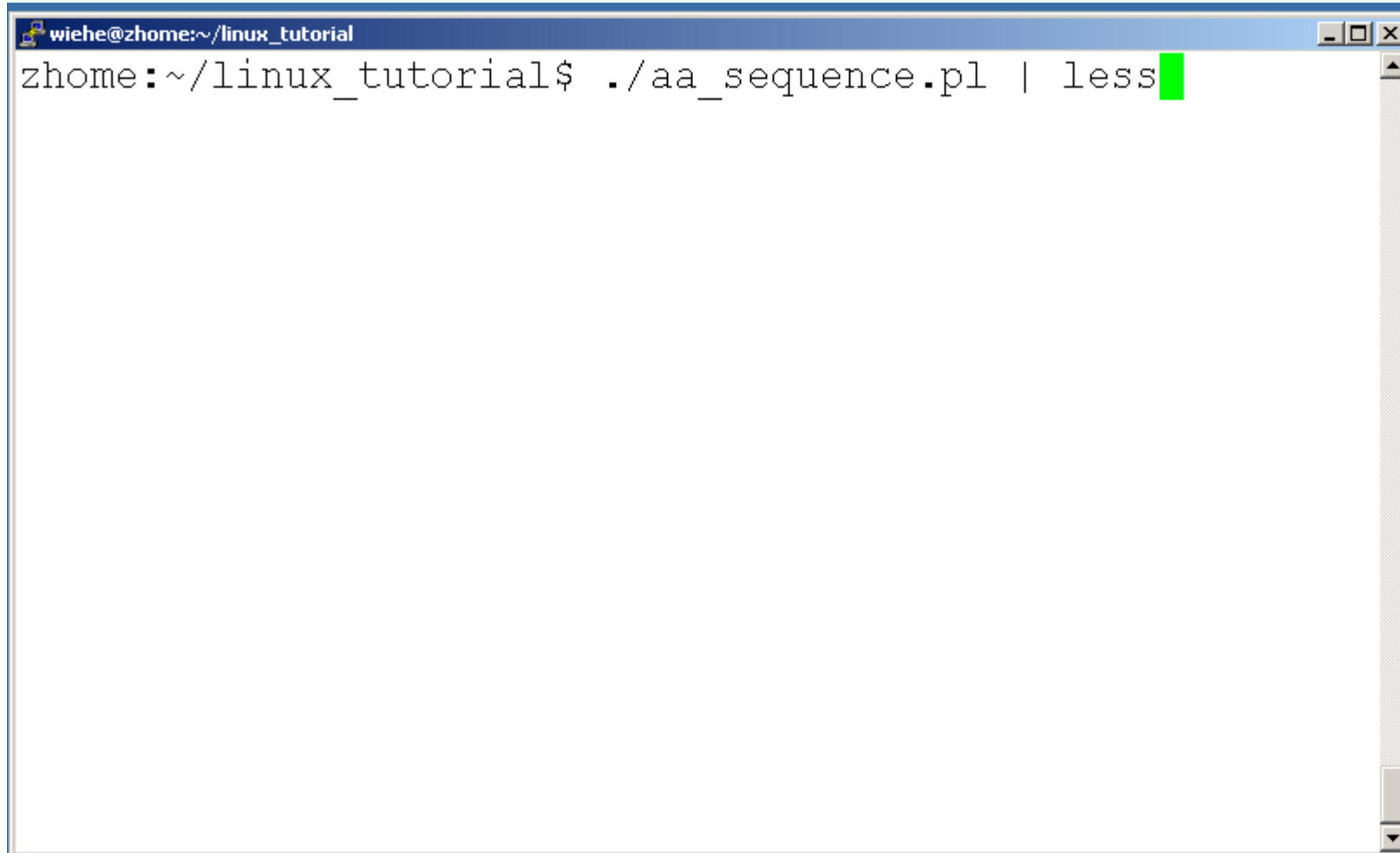
A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing the execution of the 'kill' command. The user first runs 'ps -u wiehe' to list processes. Then, they run 'kill -9 1255' to terminate the process with PID 1255. The terminal shows '[1]+ Killed ./ACTG.pl' as confirmation. Finally, they run 'ps -u wiehe' again to show the updated process list.

```
wiehe@zhome:~/linux_tutorial$ ps -u wiehe
  PID TTY          TIME CMD
 1194 ?            00:00:00 sshd
 1196 pts/2        00:00:00 bash
 1255 pts/2        00:00:01 ACTG.pl
 1287 pts/2        00:00:00 ps
zhome:~/linux_tutorial$ kill -9 1255
[1]+  Killed                  ./ACTG.pl
zhome:~/linux_tutorial$ ps -u wiehe
  PID TTY          TIME CMD
 1194 ?            00:00:00 sshd
 1196 pts/2        00:00:00 bash
 1289 pts/2        00:00:00 ps
zhome:~/linux_tutorial$
```

# Input/Output Redirection (“piping”)

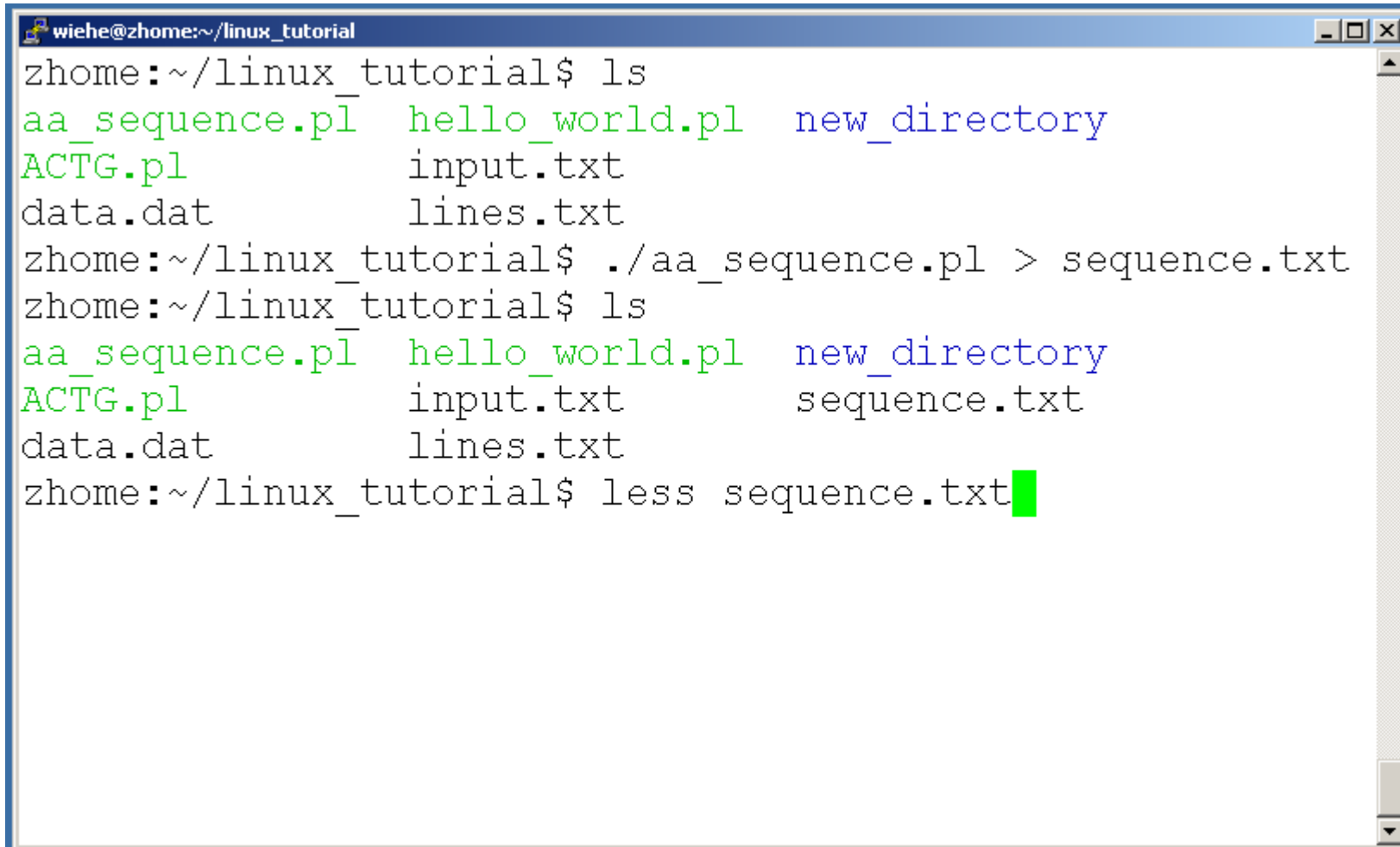
- Programs can output to other programs
- Called “piping”
- “program\_a | program\_b”
  - program\_a’s output becomes program\_b’s input
- “program\_a > file.txt”
  - program\_a’s output is written to a file called “file.txt”
- “program\_a < input.txt”
  - program\_a gets its input from a file called “input.txt”

# A few examples of piping

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal area is white and shows the command 'zhome:~/linux\_tutorial\$ ./aa\_sequence.pl | less' followed by a green cursor. The window has standard Linux window controls (minimize, maximize, close) in the top right corner and a vertical scrollbar on the right side.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ./aa_sequence.pl | less
```

# A few examples of piping

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The window shows a series of commands and their outputs. The first command is 'ls', which lists files: 'aa\_sequence.pl', 'hello\_world.pl', 'new\_directory', 'ACTG.pl', 'input.txt', 'data.dat', and 'lines.txt'. The second command is './aa\_sequence.pl > sequence.txt'. The third command is another 'ls', which now includes 'sequence.txt' in the list. The fourth command is 'less sequence.txt', followed by a green cursor block.

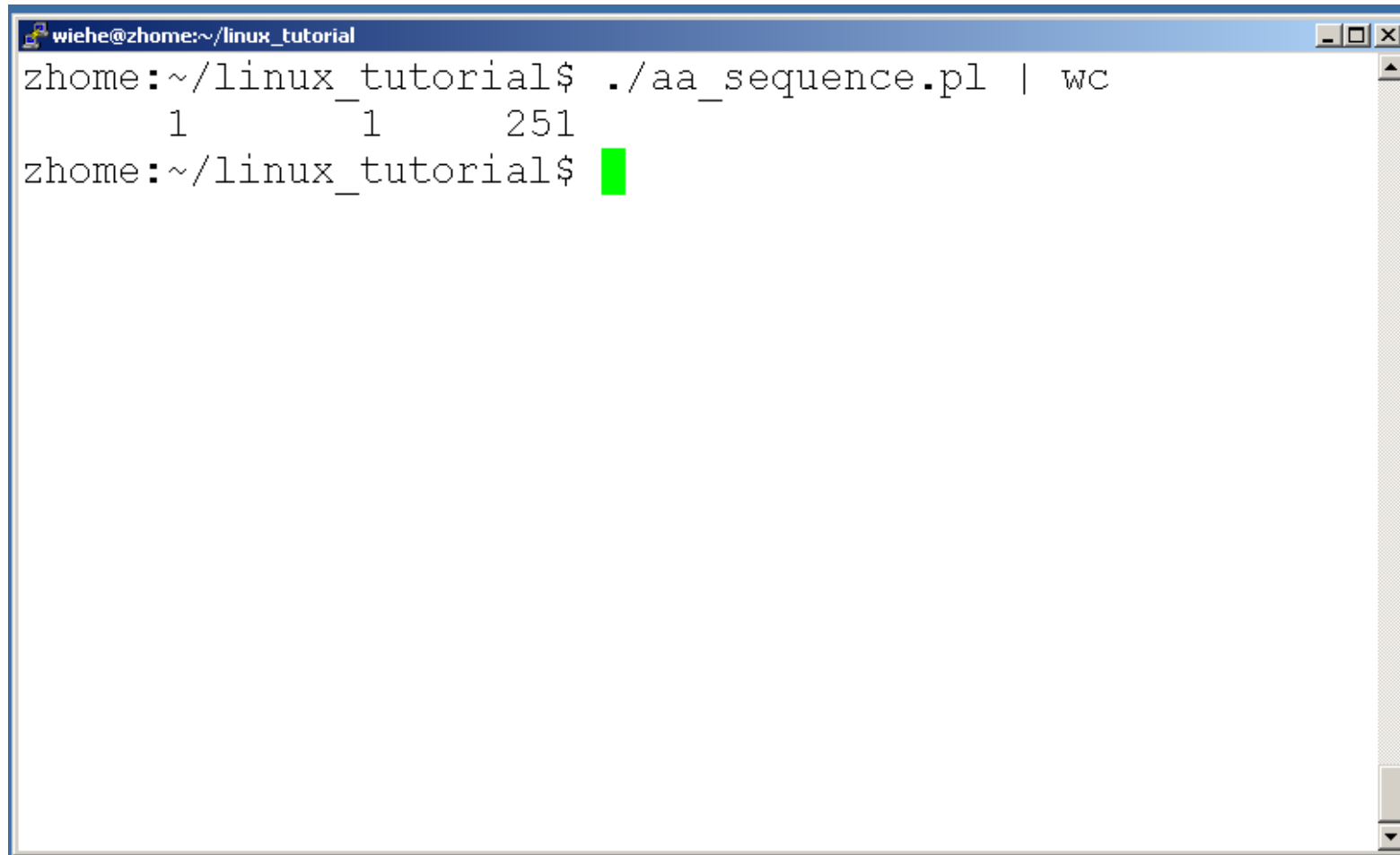
```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  hello_world.pl  new_directory
ACTG.pl        input.txt
data.dat       lines.txt
zhome:~/linux_tutorial$ ./aa_sequence.pl > sequence.txt
zhome:~/linux_tutorial$ ls
aa_sequence.pl  hello_world.pl  new_directory
ACTG.pl        input.txt       sequence.txt
data.dat       lines.txt
zhome:~/linux_tutorial$ less sequence.txt
```

# Command: wc

- To count the characters, words, and lines in a file use “wc”
- The first column in the output is lines, the second is words, and the last is characters



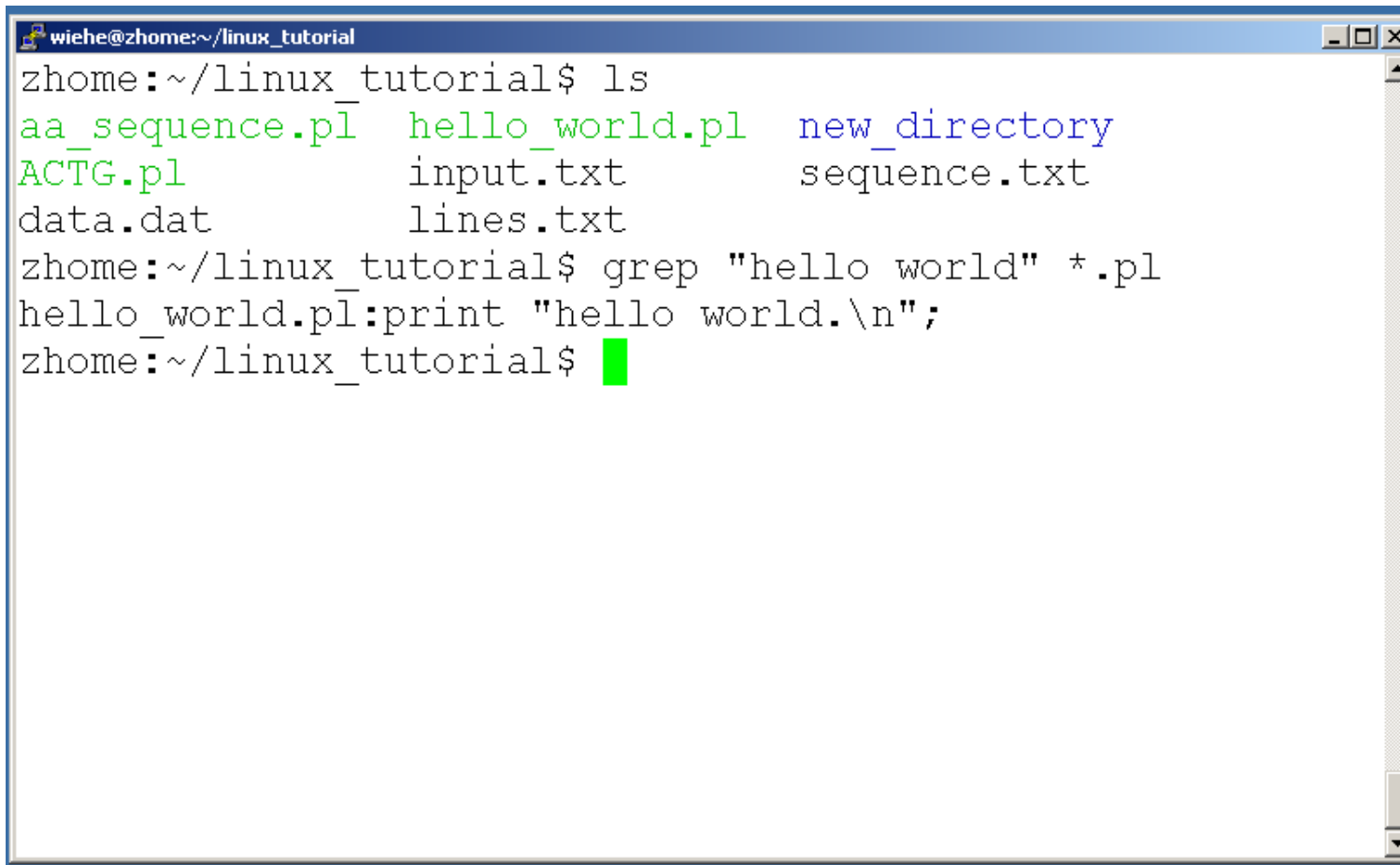
# A few examples of piping

A terminal window with a blue title bar containing the text 'wiehe@zhome:~/linux\_tutorial'. The terminal shows a command being executed and its output. The command is './aa\_sequence.pl | wc'. The output is '1 1 251'. The prompt 'zhome:~/linux\_tutorial\$' is shown twice, once before the command and once after the output.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ./aa_sequence.pl | wc
1 1 251
zhome:~/linux_tutorial$
```

# Command: grep

- To search files in a directory for a specific string use “grep”

A terminal window titled 'wiehe@zhome:~/linux\_tutorial' showing a series of commands and their outputs. The user first runs 'ls' to list files, then 'grep "hello world" \*.pl' to search for the string 'hello world' in all Perl files. The output shows that 'hello\_world.pl' contains the string.

```
wiehe@zhome:~/linux_tutorial
zhome:~/linux_tutorial$ ls
aa_sequence.pl  hello_world.pl  new_directory
ACTG.pl        input.txt       sequence.txt
data.dat       lines.txt
zhome:~/linux_tutorial$ grep "hello world" *.pl
hello_world.pl:print "hello world.\n";
zhome:~/linux_tutorial$
```

# Command: diff

- To compare to files for differences use “diff”
  - Try: `diff /dev/null hello.txt`
  - `/dev/null` is a special address -- it is always empty, and anything moved there is deleted

# Retrospectiva – De-brief

## QUÉ HICIMOS BIEN!

Buen repaso

Descripcion de  
componentes

Drivers antes de  
asterisk

Comandos  
basicos

Muchos temas,  
entendibles.

Explicación de  
licencias.

## QUÉ PODEMOS MEJORAR

Falta la practica

Material

No tanta teoría...

Tener info del  
examen.