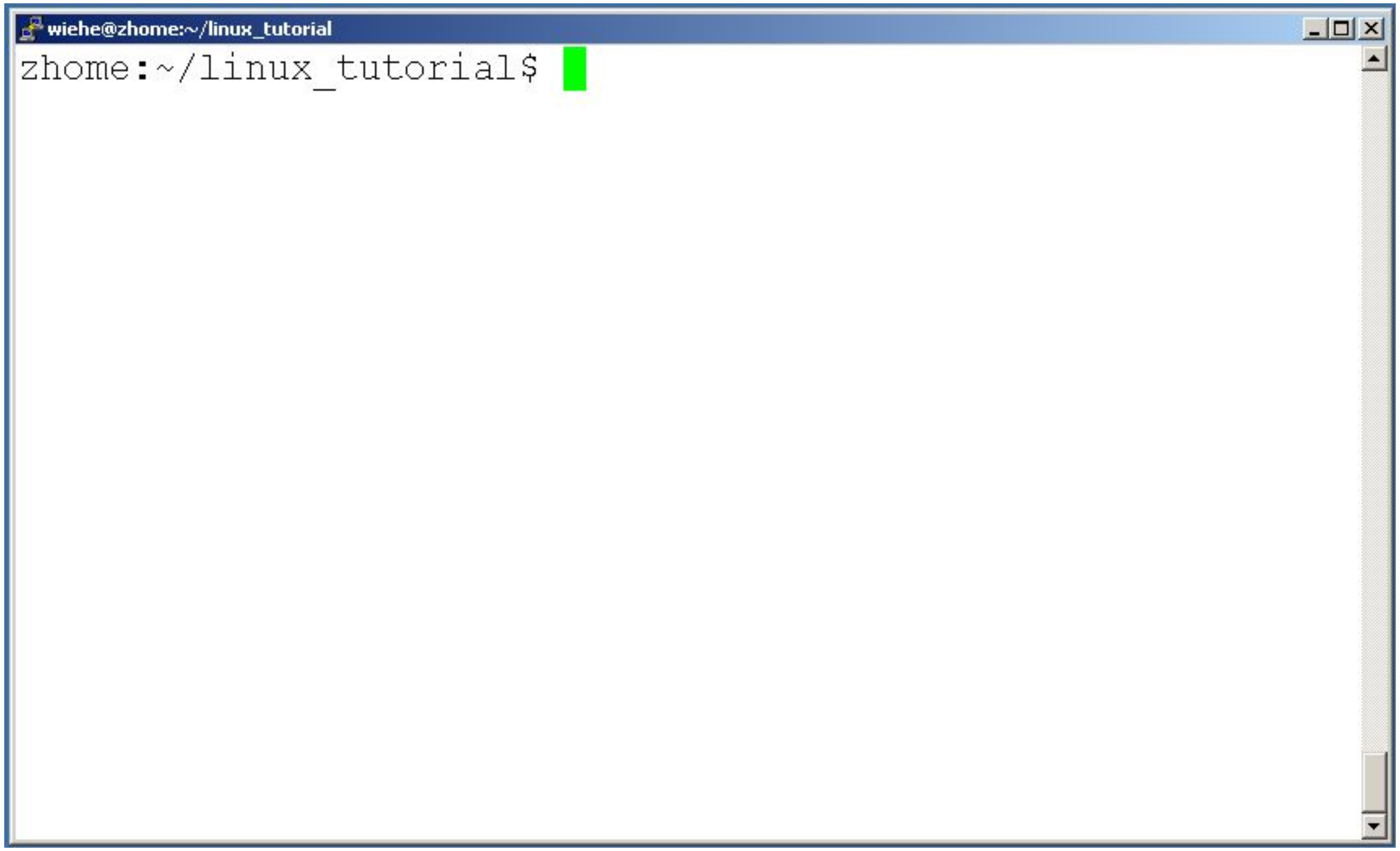
BASIC UNIX/LINUX COMMANDS

OS CLASS IB

CONNECTING TO A UNIX/LINUX SYSTEM

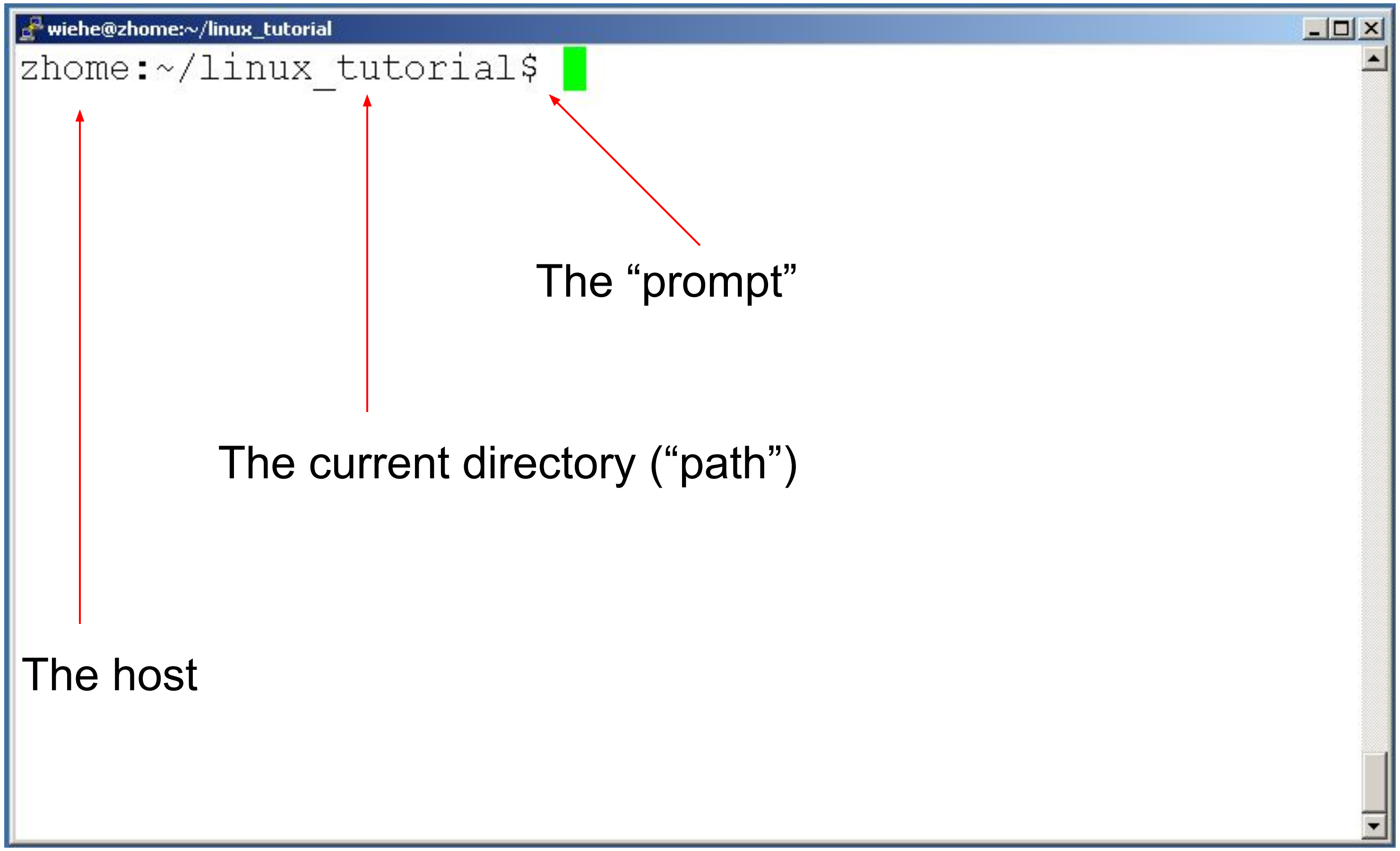
Open up a terminal:



```
zhome:~/linux_tutorial$
```

CONNECTING TO A UNIX/LINUX SYSTEM

Open up a terminal:



```
zhome:~/linux_tutorial$
```

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

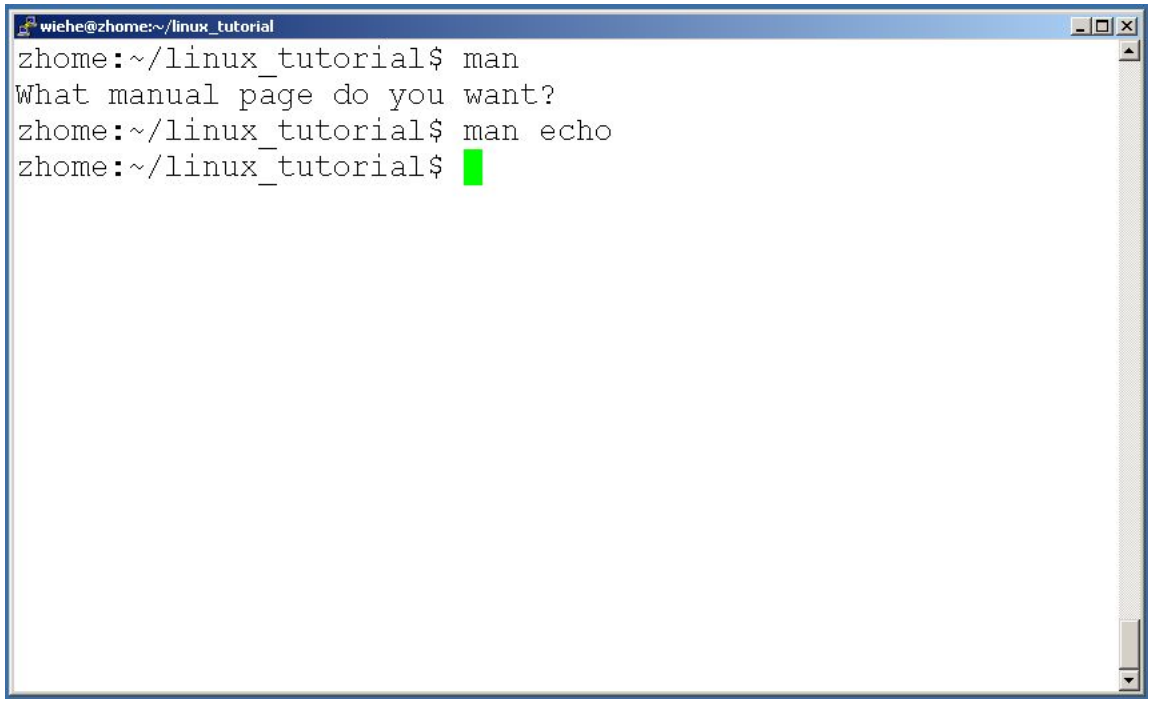
- The shell communicates with the internal part of the operating system called the **kernel**
- The most popular shells are: tcsh, csh, korn, and bash
- The differences are most times subtle
- For this tutorial, we are using bash

Shell commands are **CASE SENSITIVE!**

HELP!

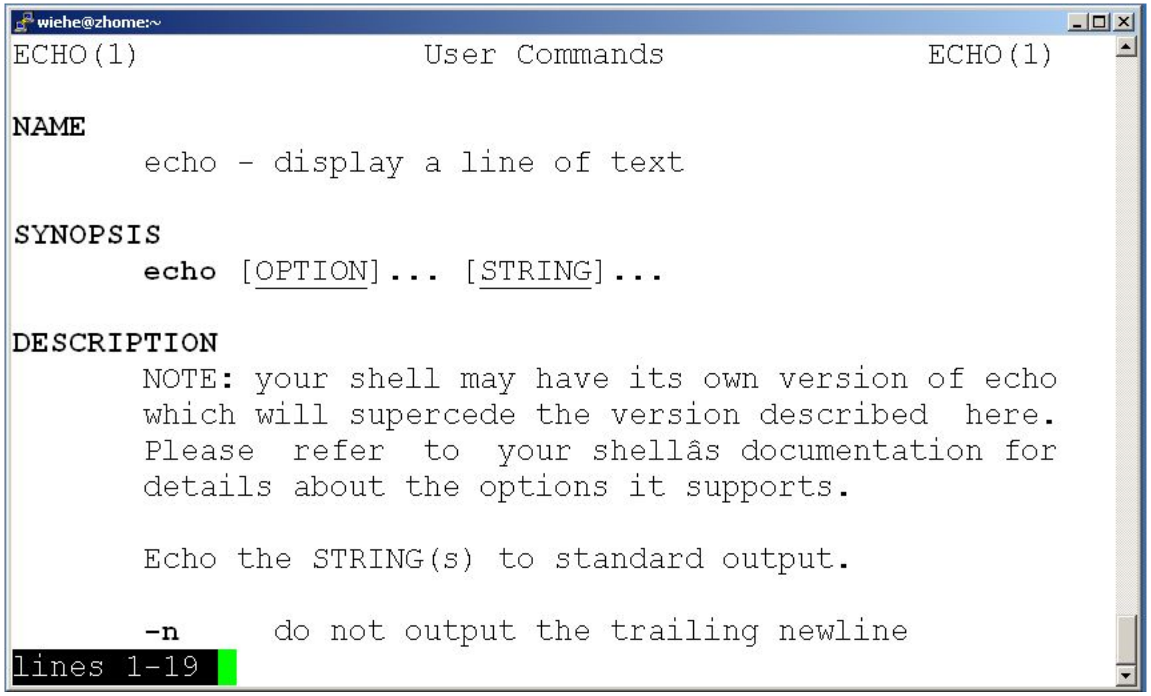
Whenever you need help with a command type “man” and the command name

HELP!



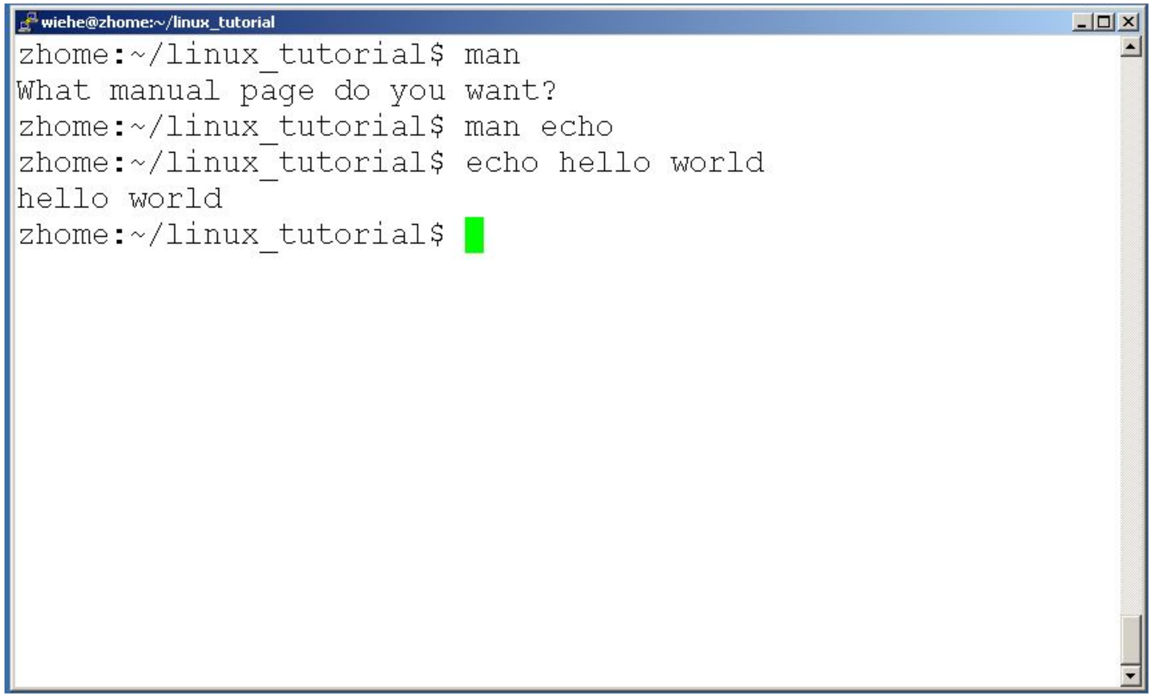
```
zhome:~/linux_tutorial$ man
What manual page do you want?
zhome:~/linux_tutorial$ man echo
zhome:~/linux_tutorial$
```

HELP!



```
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
```

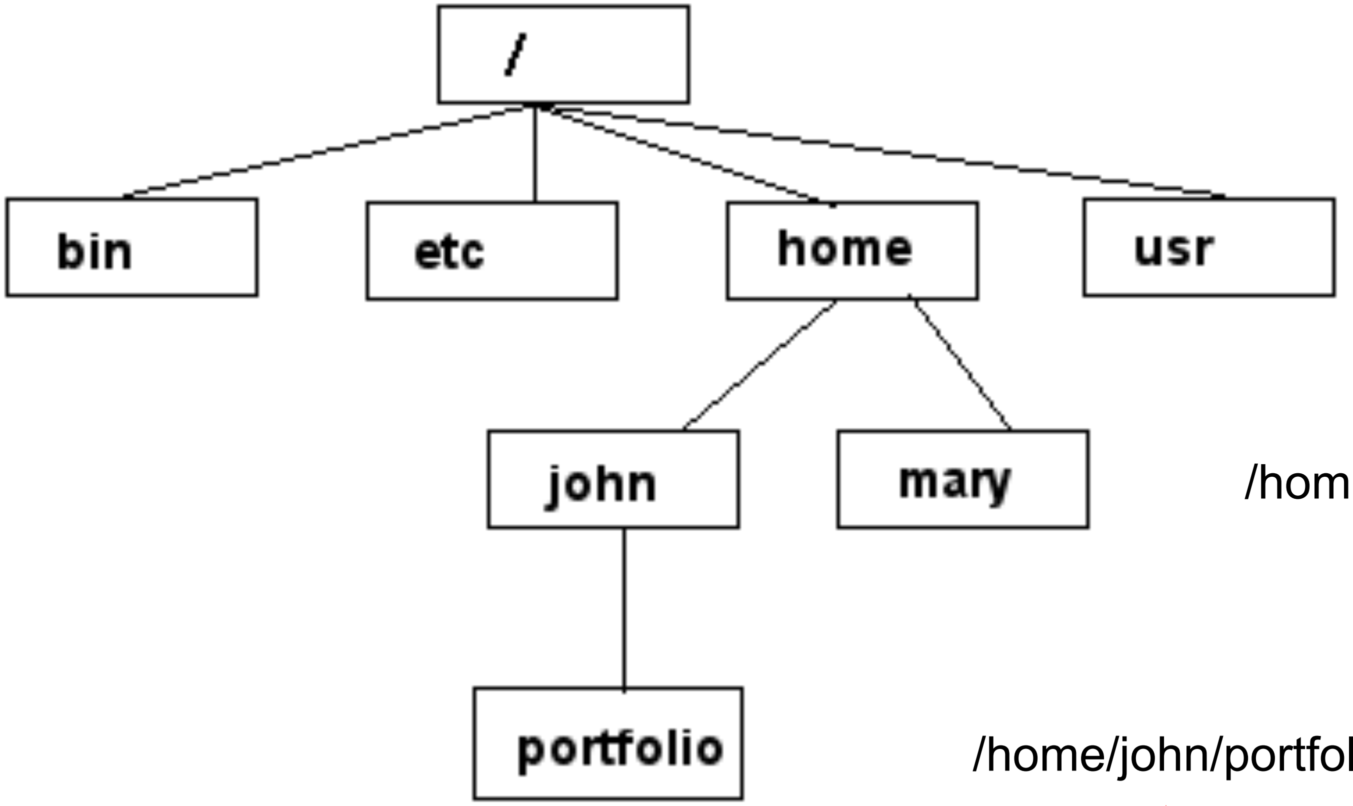
HELP!



```
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!**

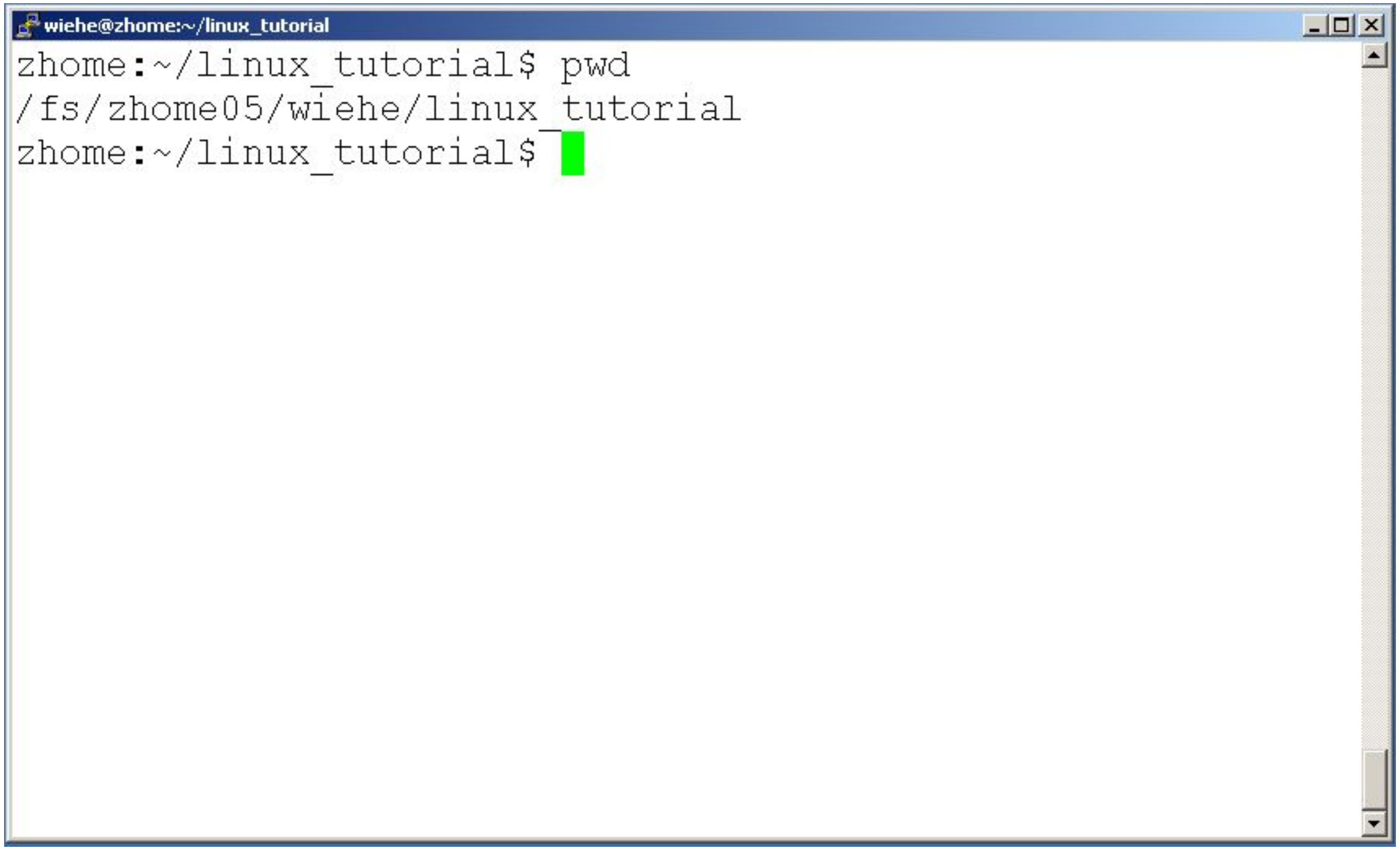


/home/mary/

/home/john/portfolio/

The Path

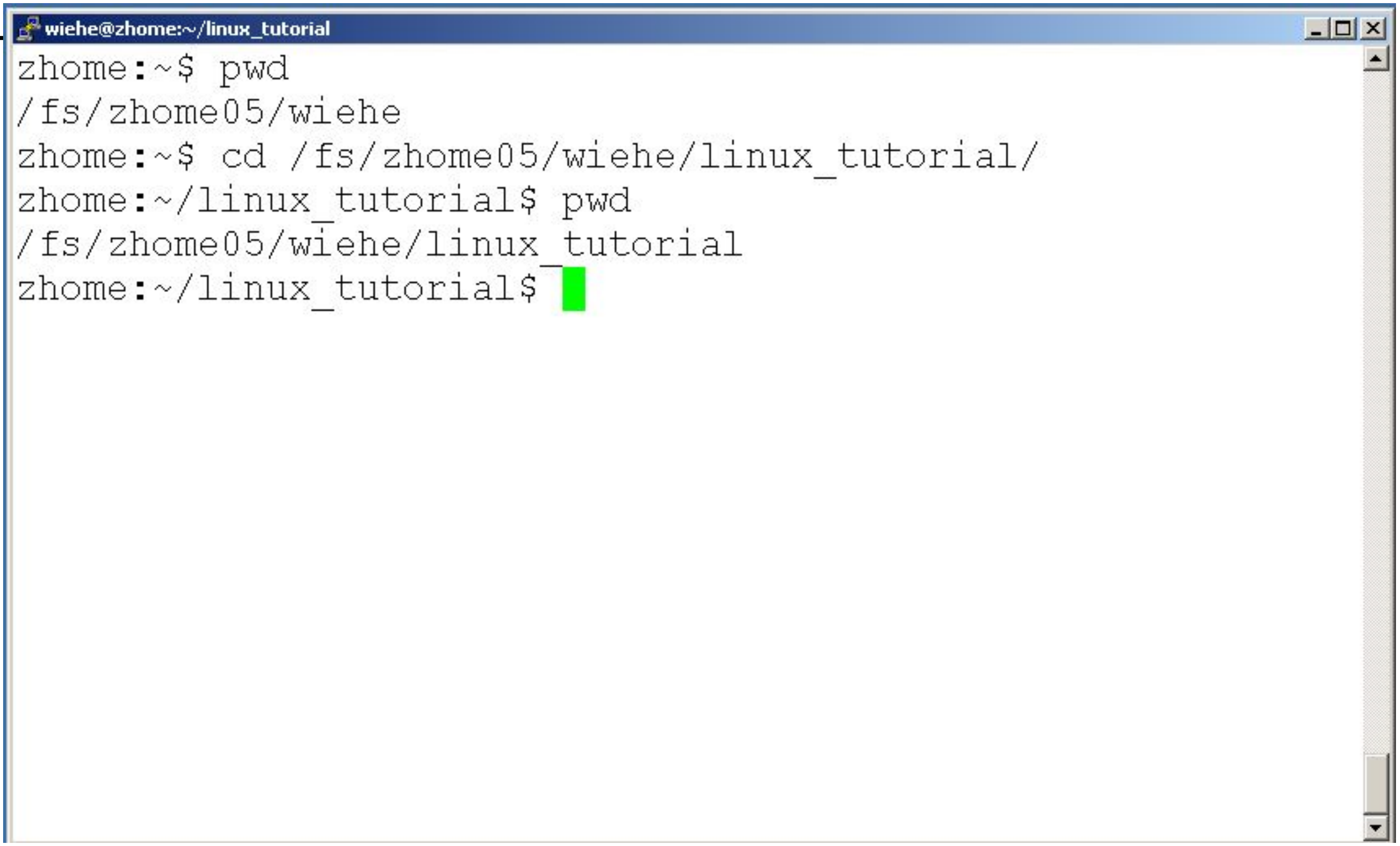
COMMAND: PWD



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

COMMAND: CD

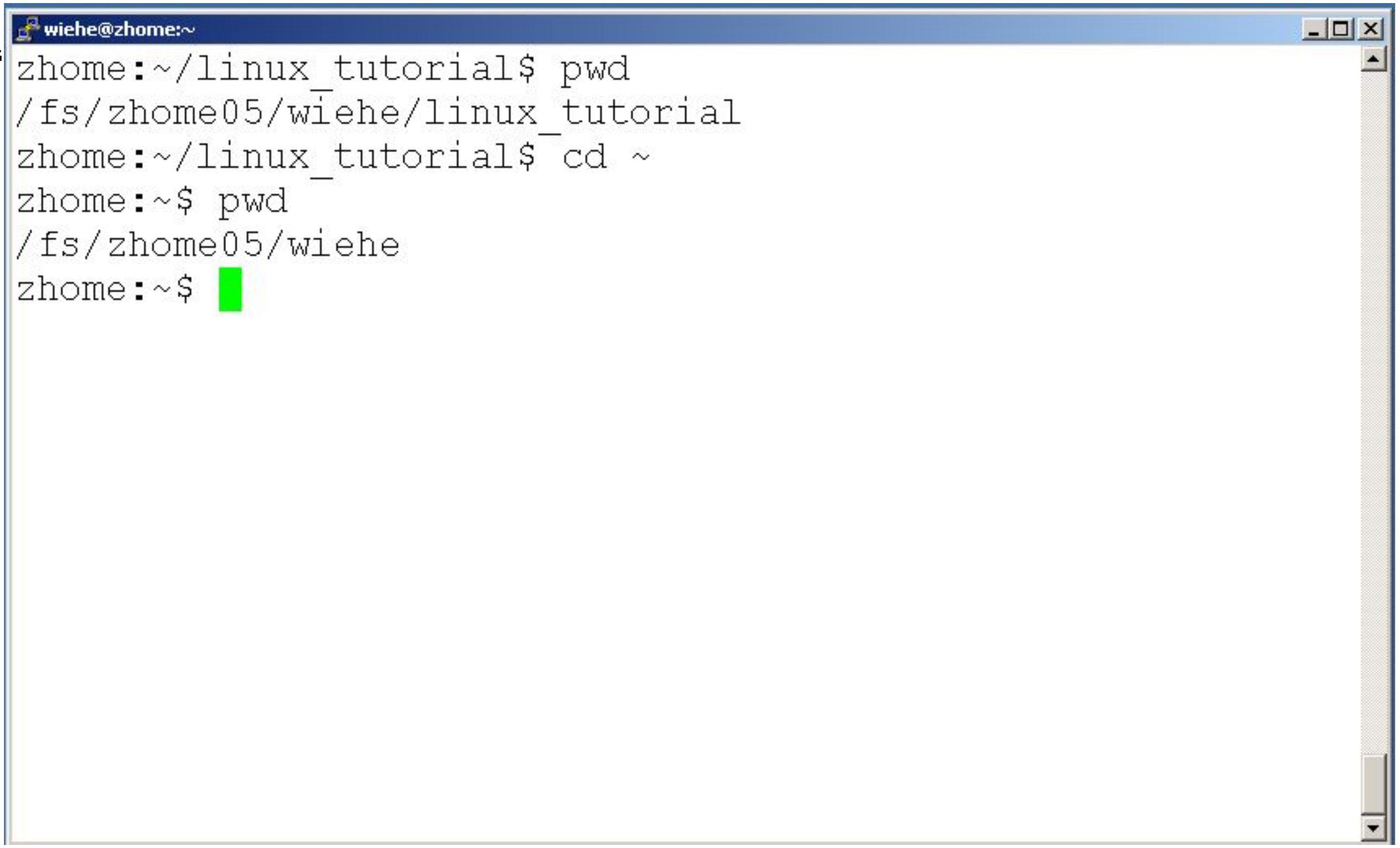
T



```
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

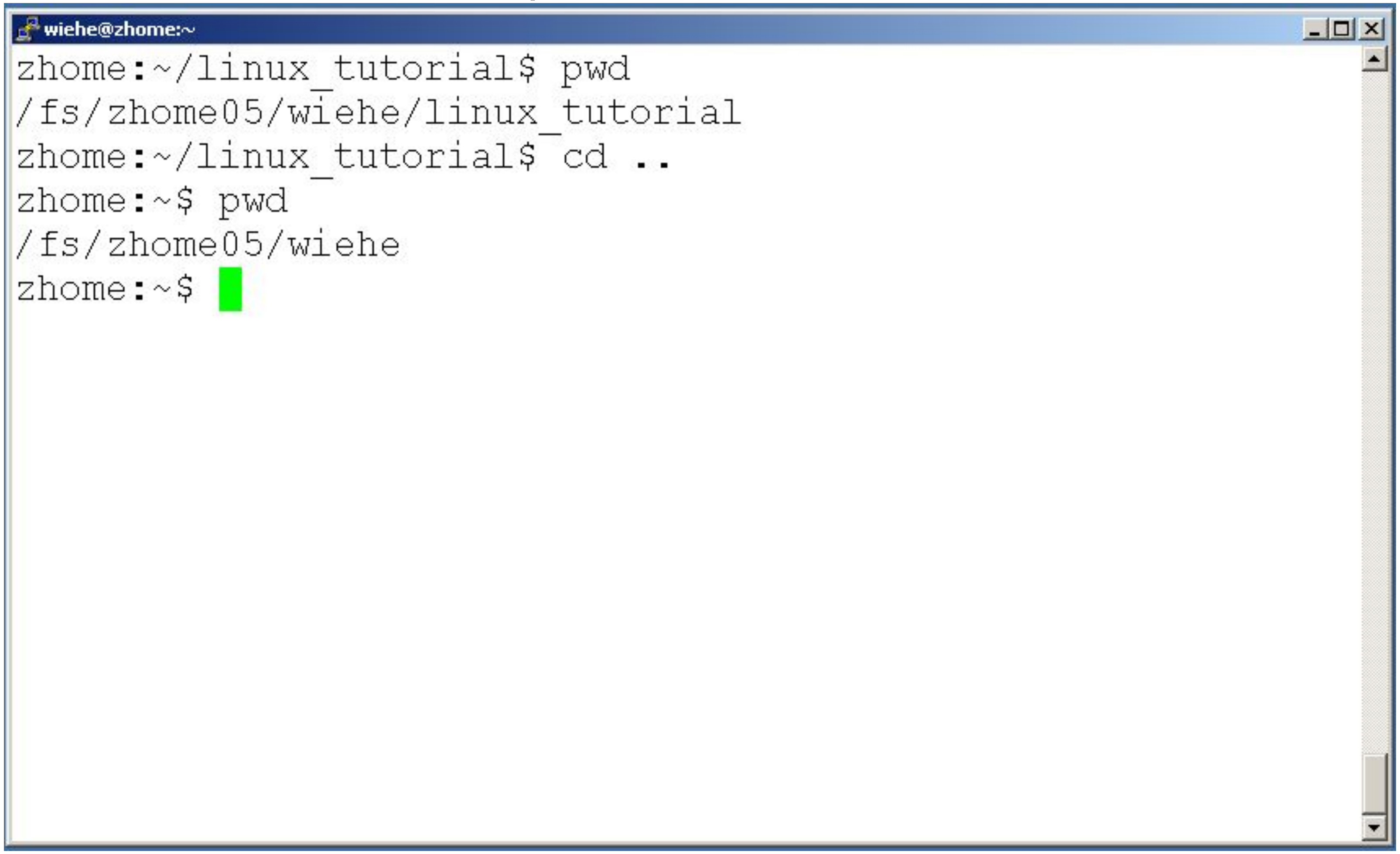
“



```
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

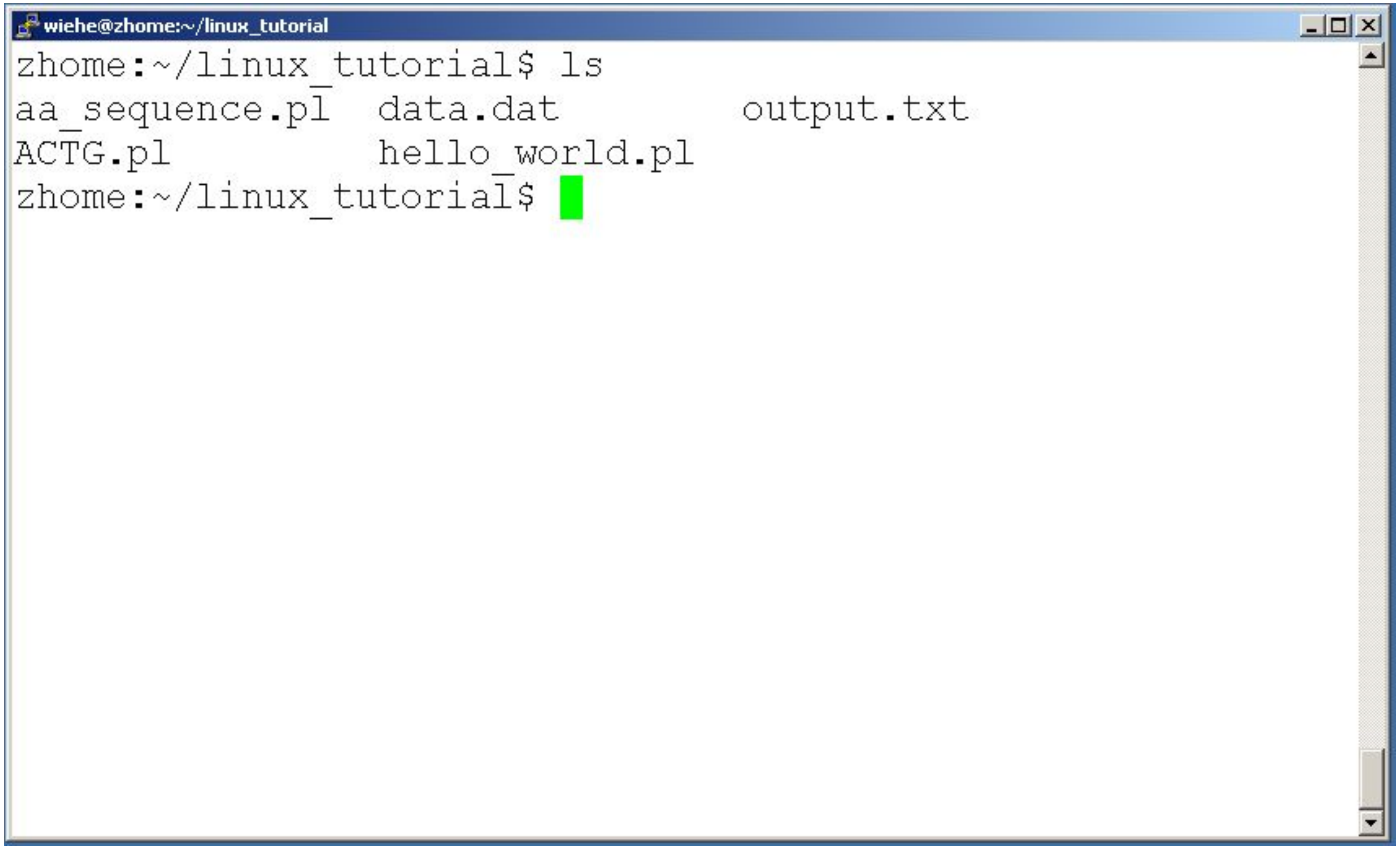


wiehe@zhome:~

```
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”


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

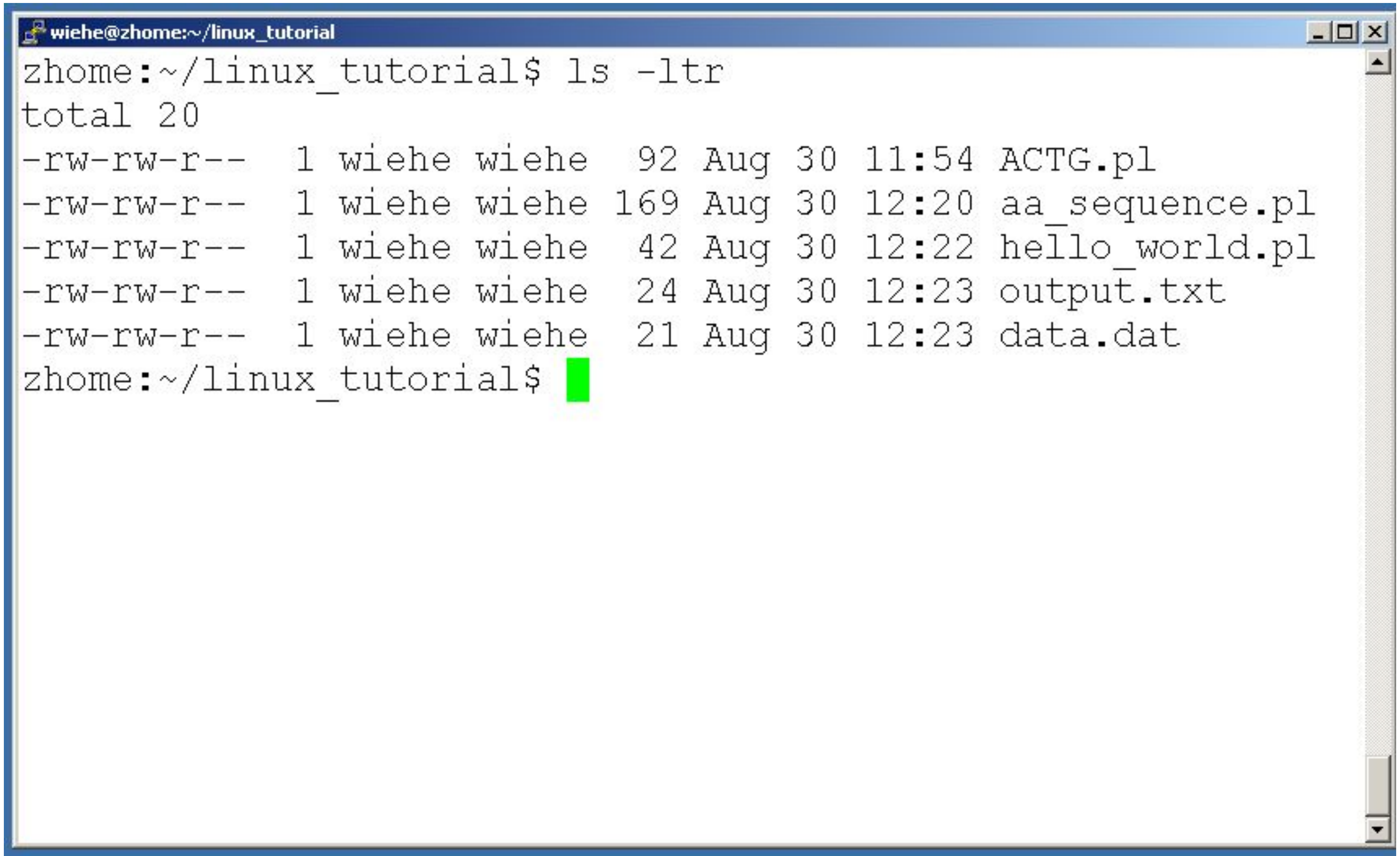
- -l long list (displays lots of info)
- -t sort by modification time
- -S sort by size
- -h list file sizes in human readable format
- -r reverse the order
- -a show all files hidden files

“man ls” for more options

Options can be combined: “ls -ltr”

COMMAND: LS -LTR

List files by time in reverse order with long listing

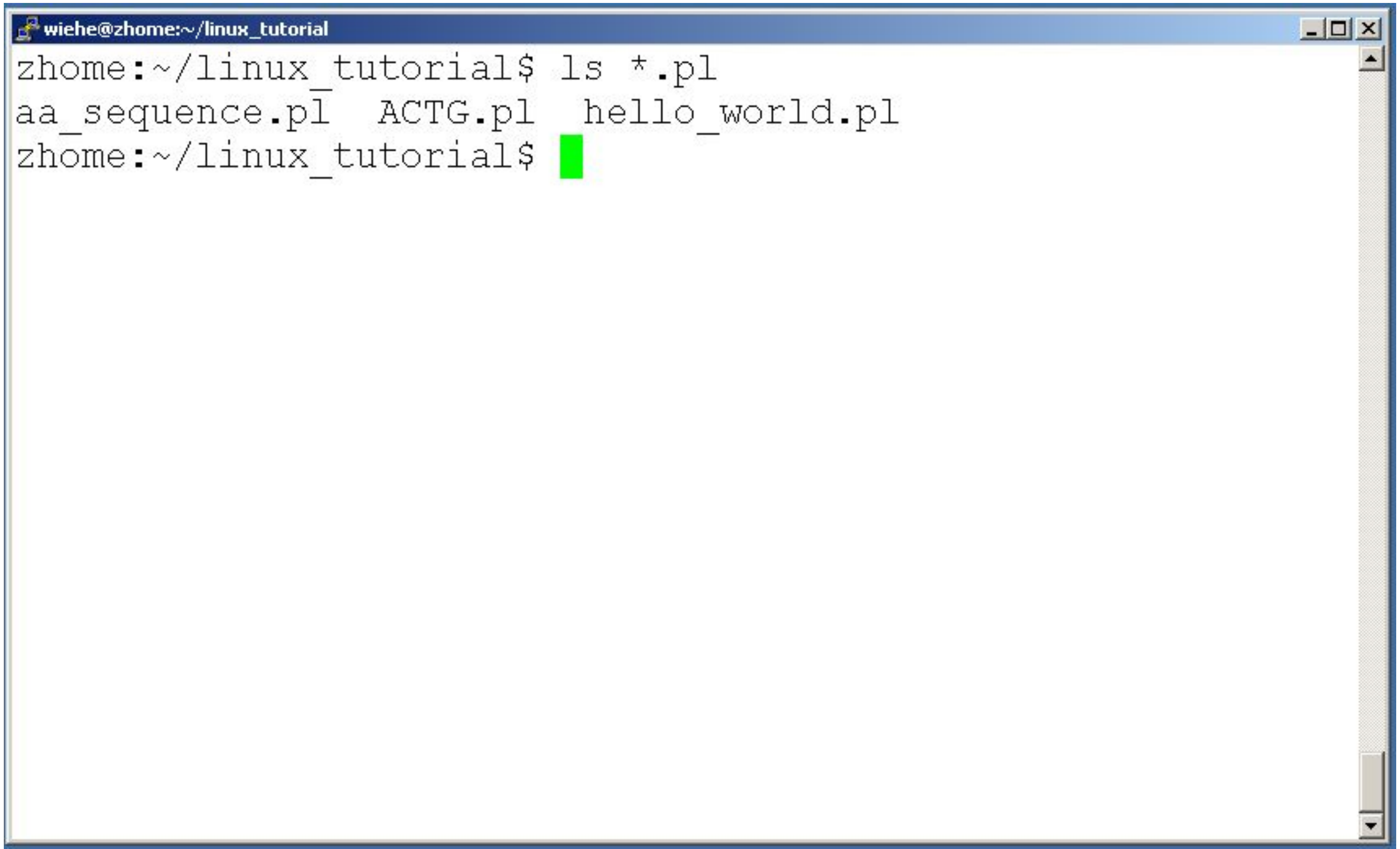


wiehe@zhome:~/linux_tutorial

```
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: *

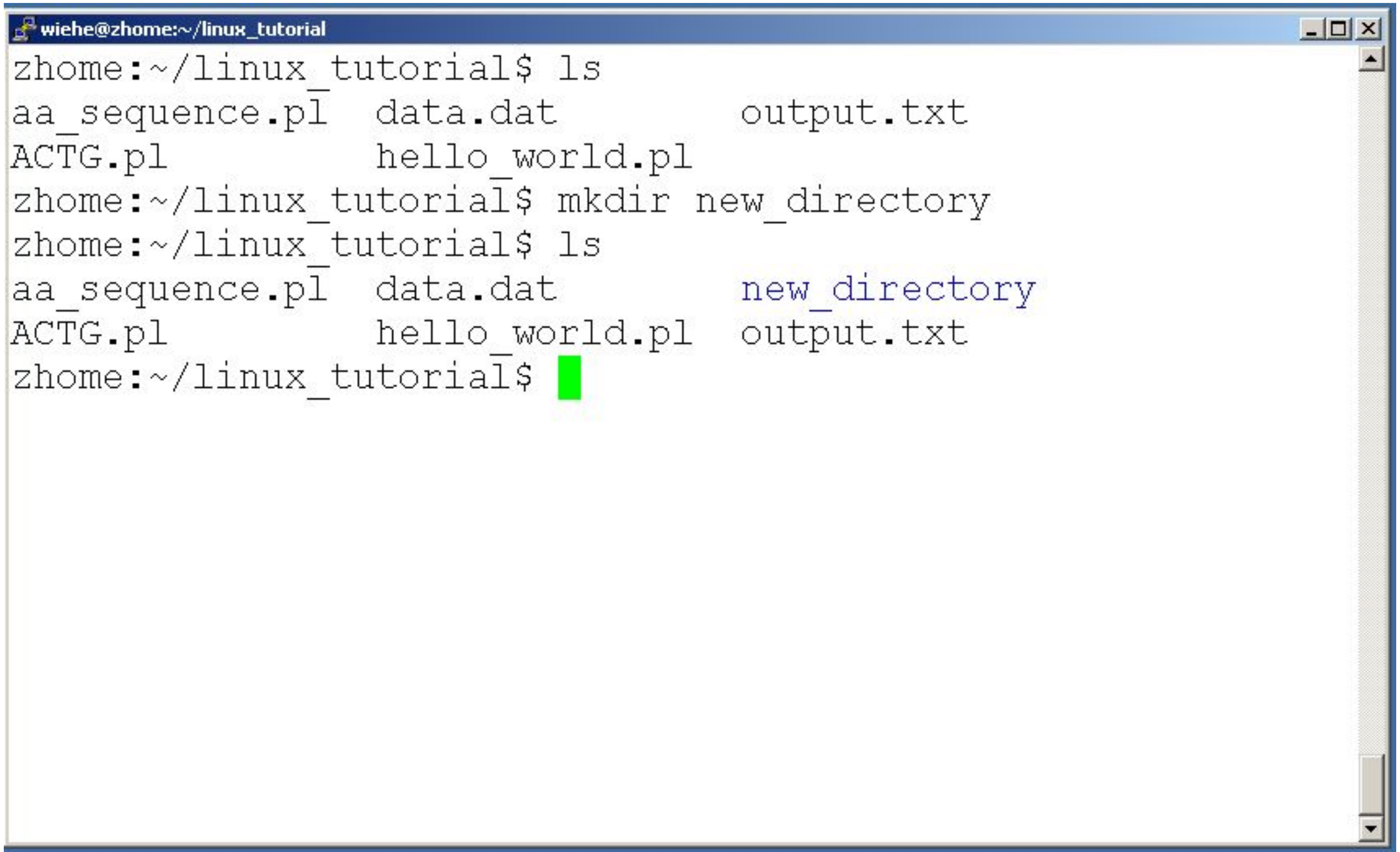
"*" can be used as a wildcard in unix/linux


wiehe@zhome:~/linux_tutorial

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

COMMAND: MKDIR

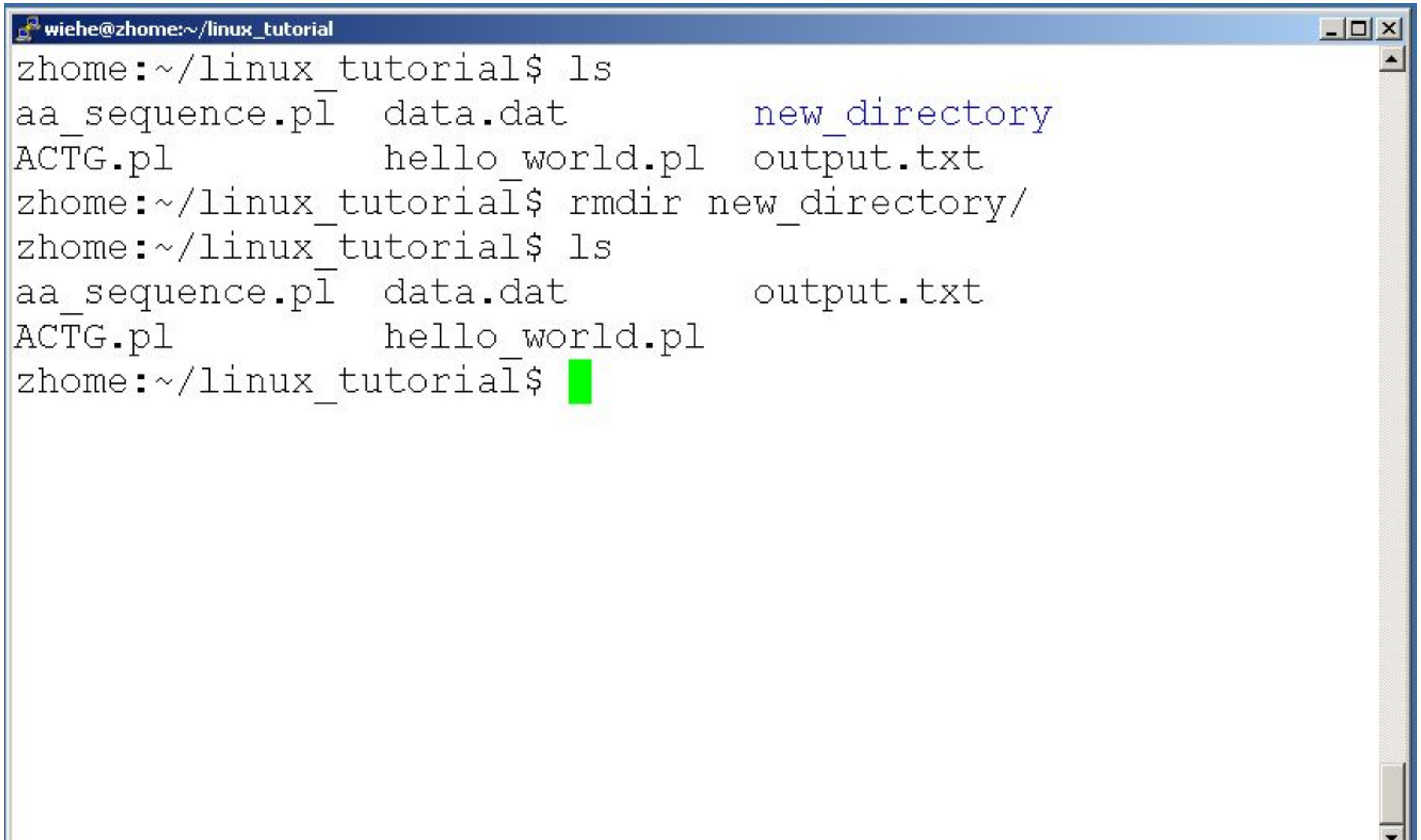
To create a new directory use "mkdir"



```
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 and 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

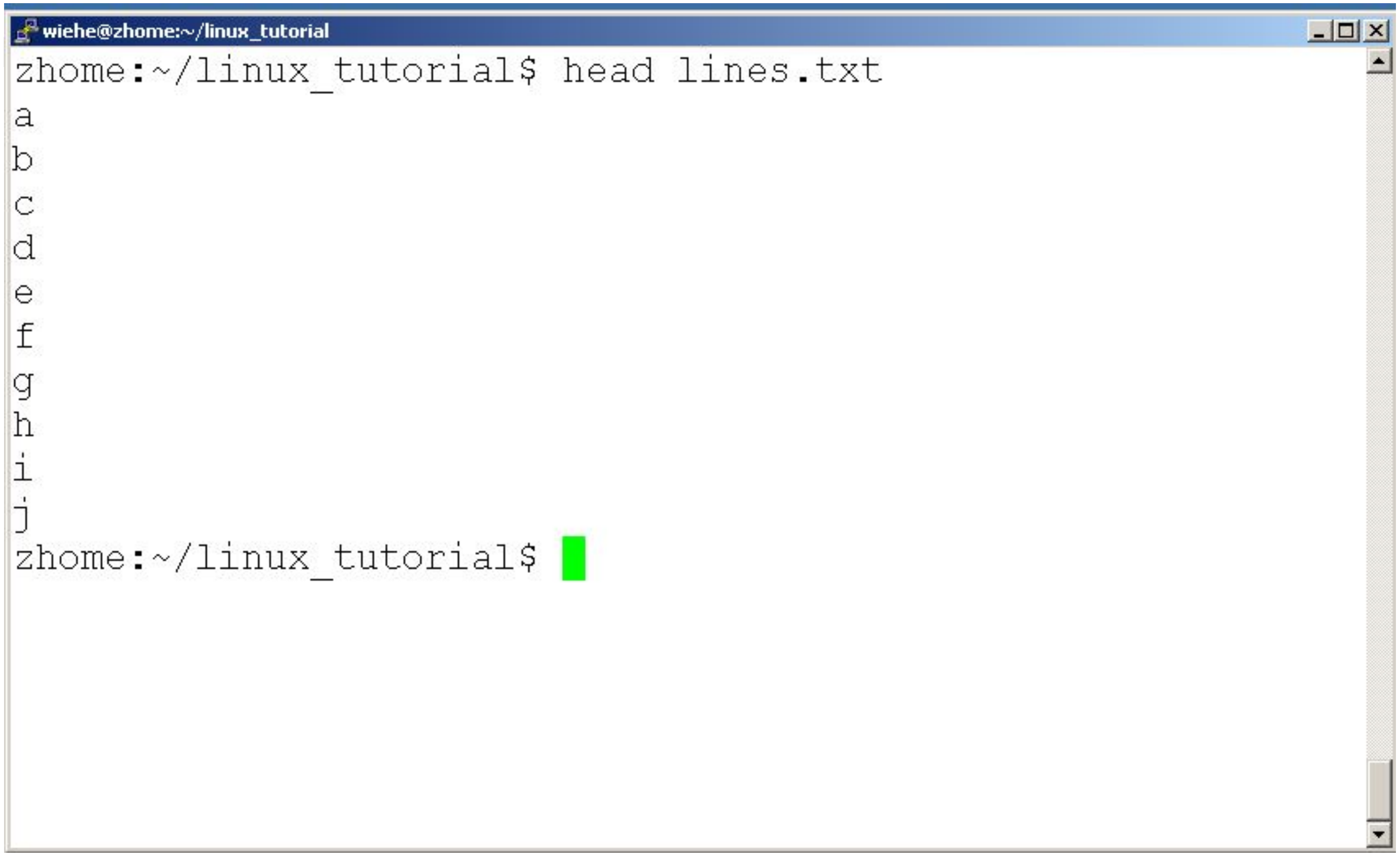
"head" displays the top part of a file

By default it shows the first 10 lines

-n option allows you to change that

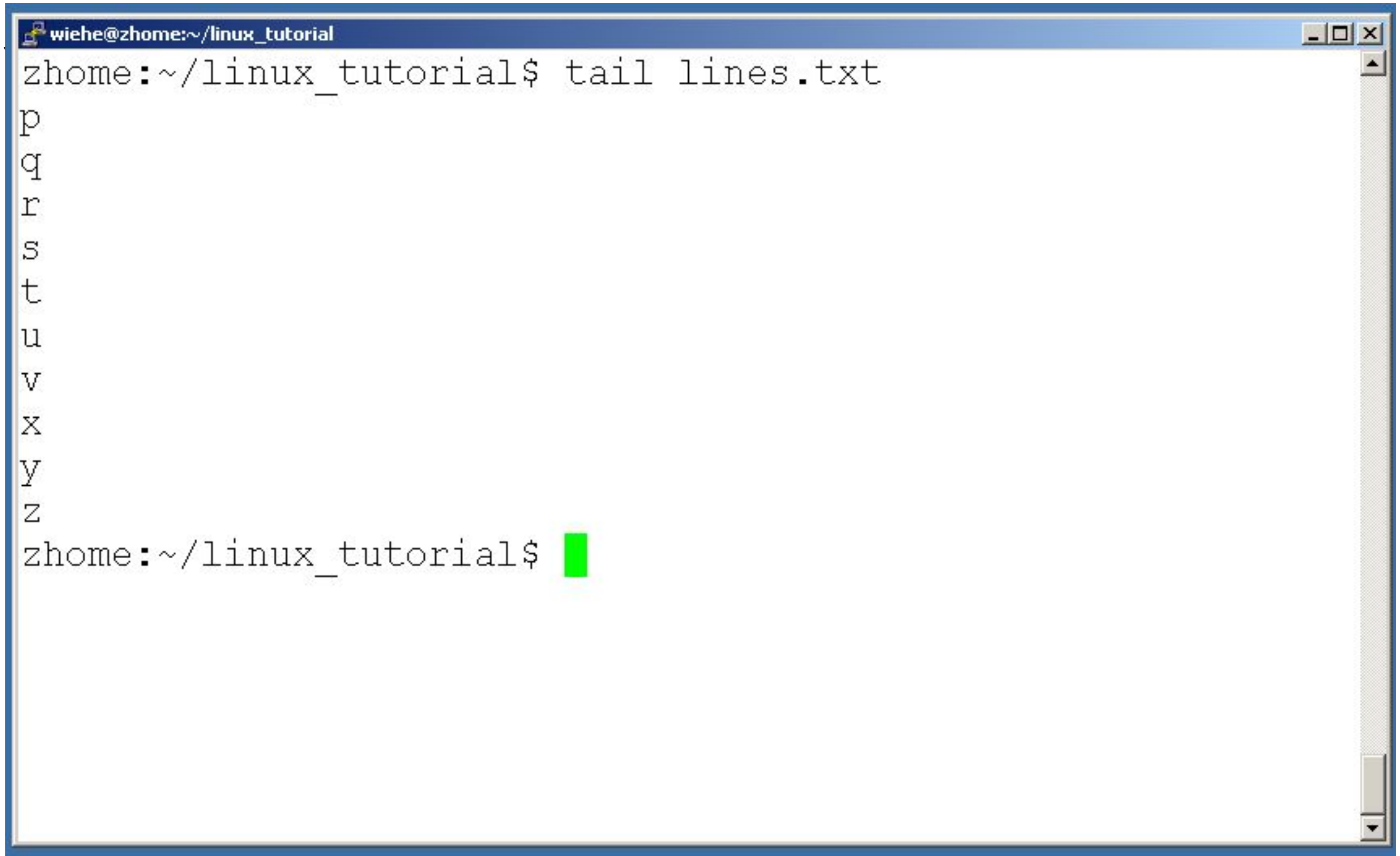
"head -n50 file.txt" displays the first 50 lines of file.txt

COMMAND: HEAD


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



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

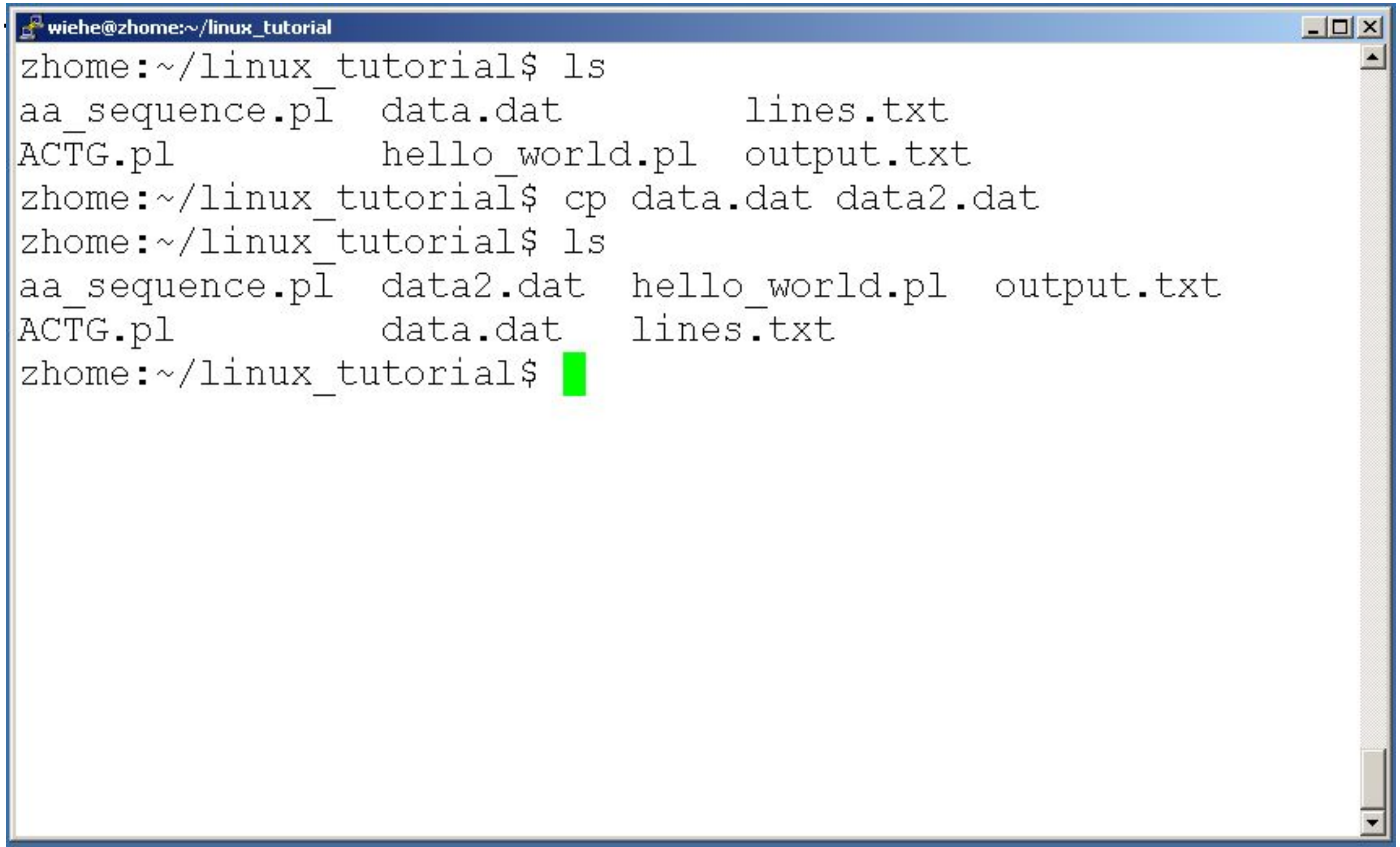
FILE COMMANDS

Copying a file: cp

Move or rename a file: mv

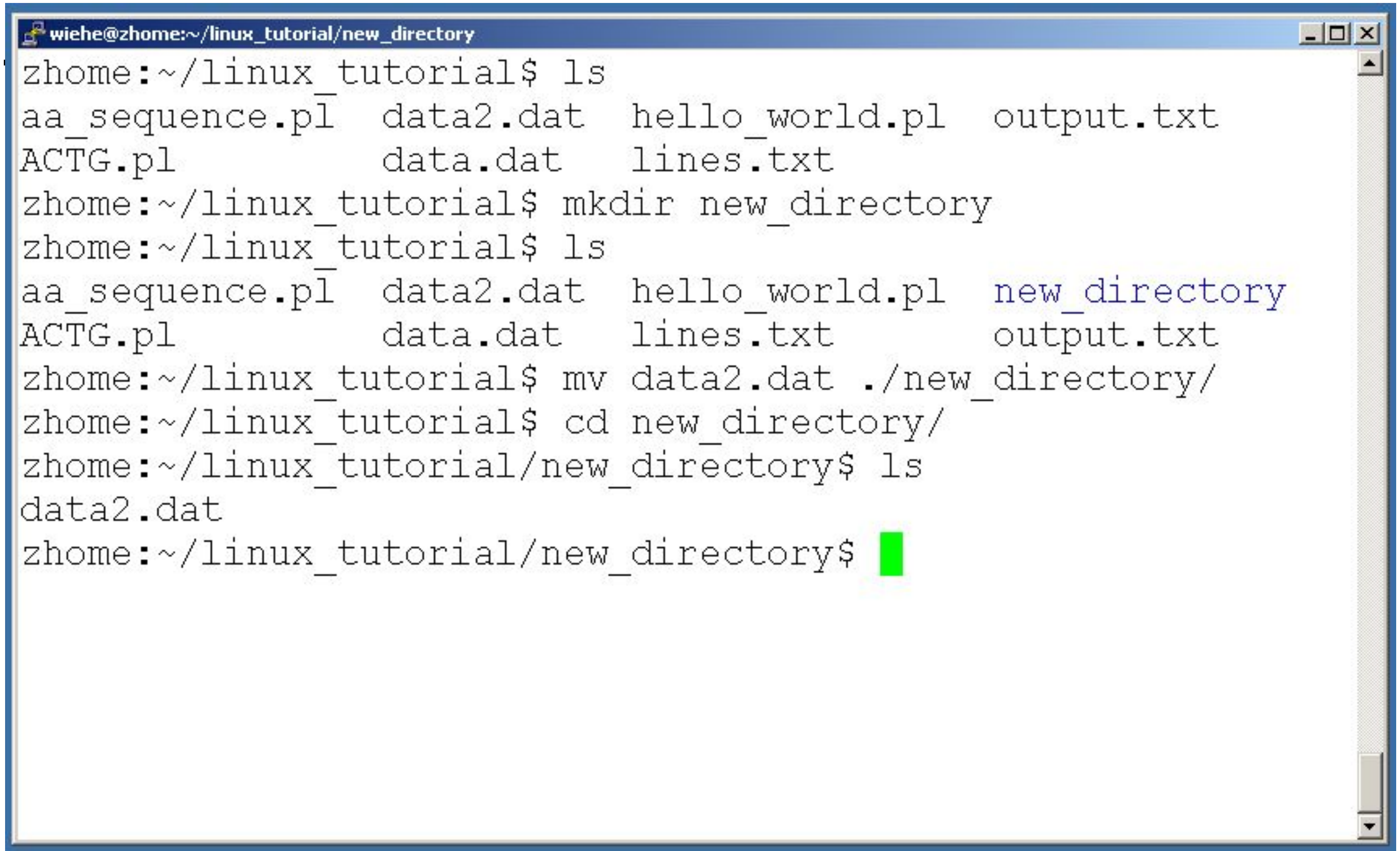
Remove a file: rm

COMMAND: CP



```
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$ 
```

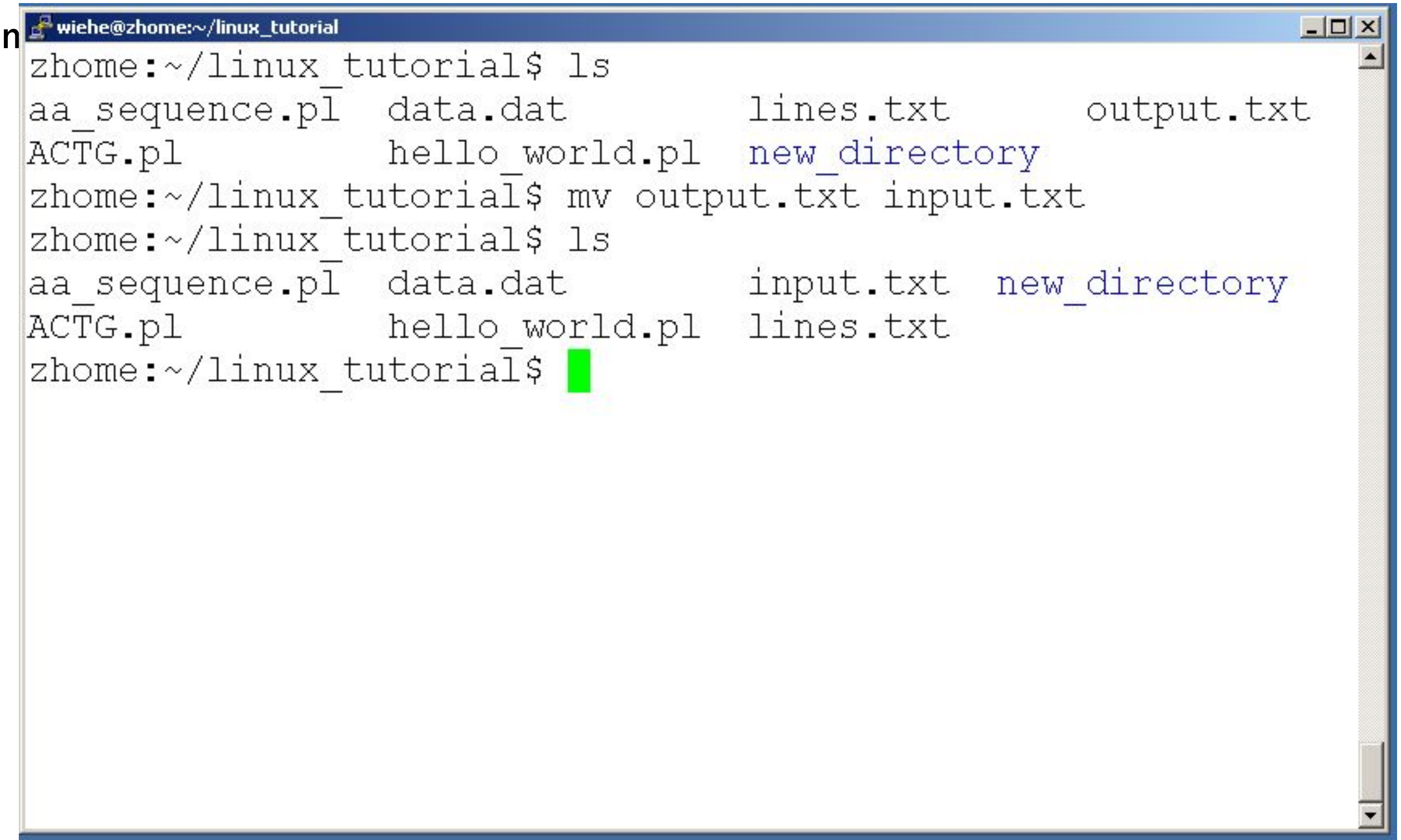
COMMAND: MV


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

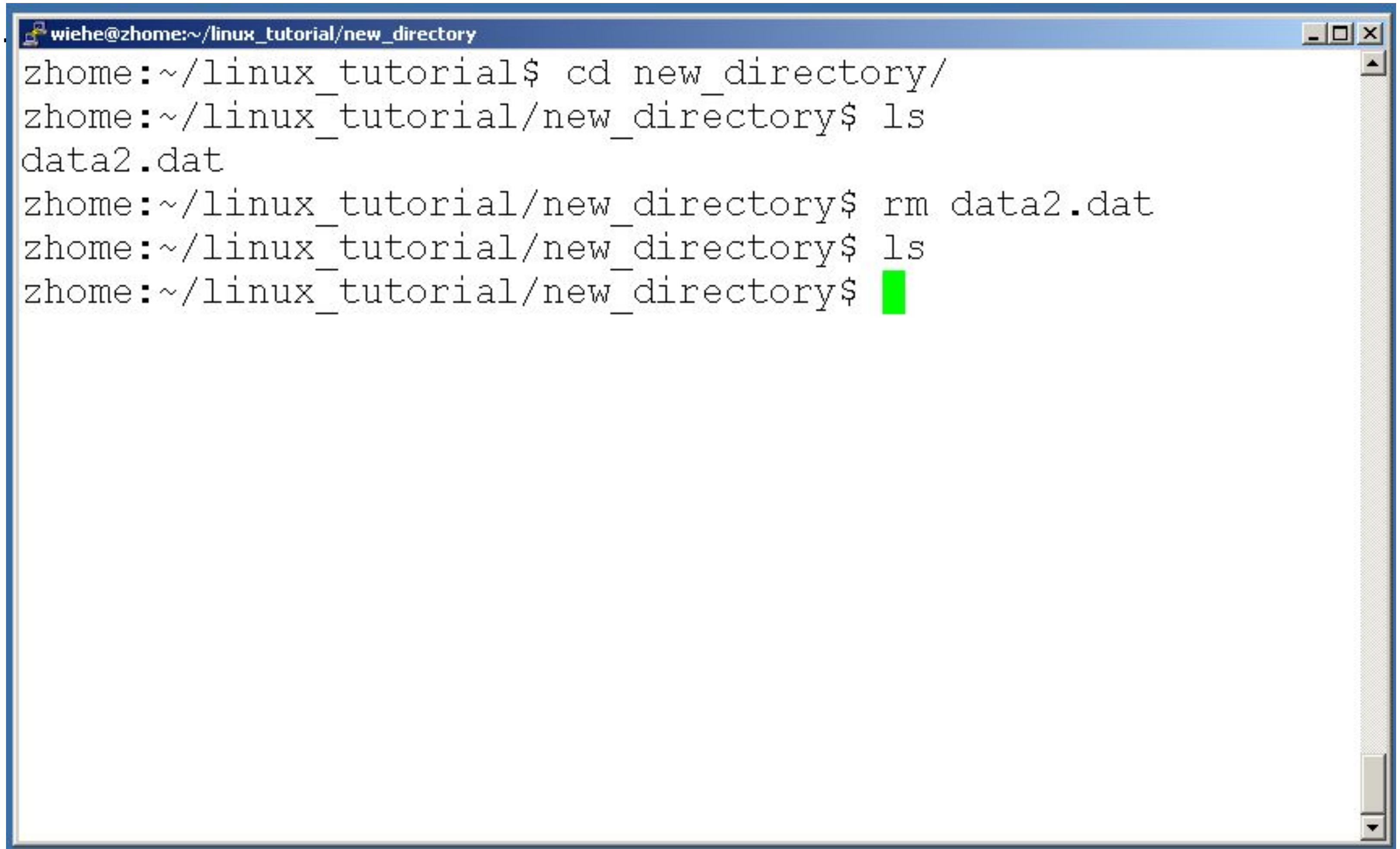
n



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



```
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

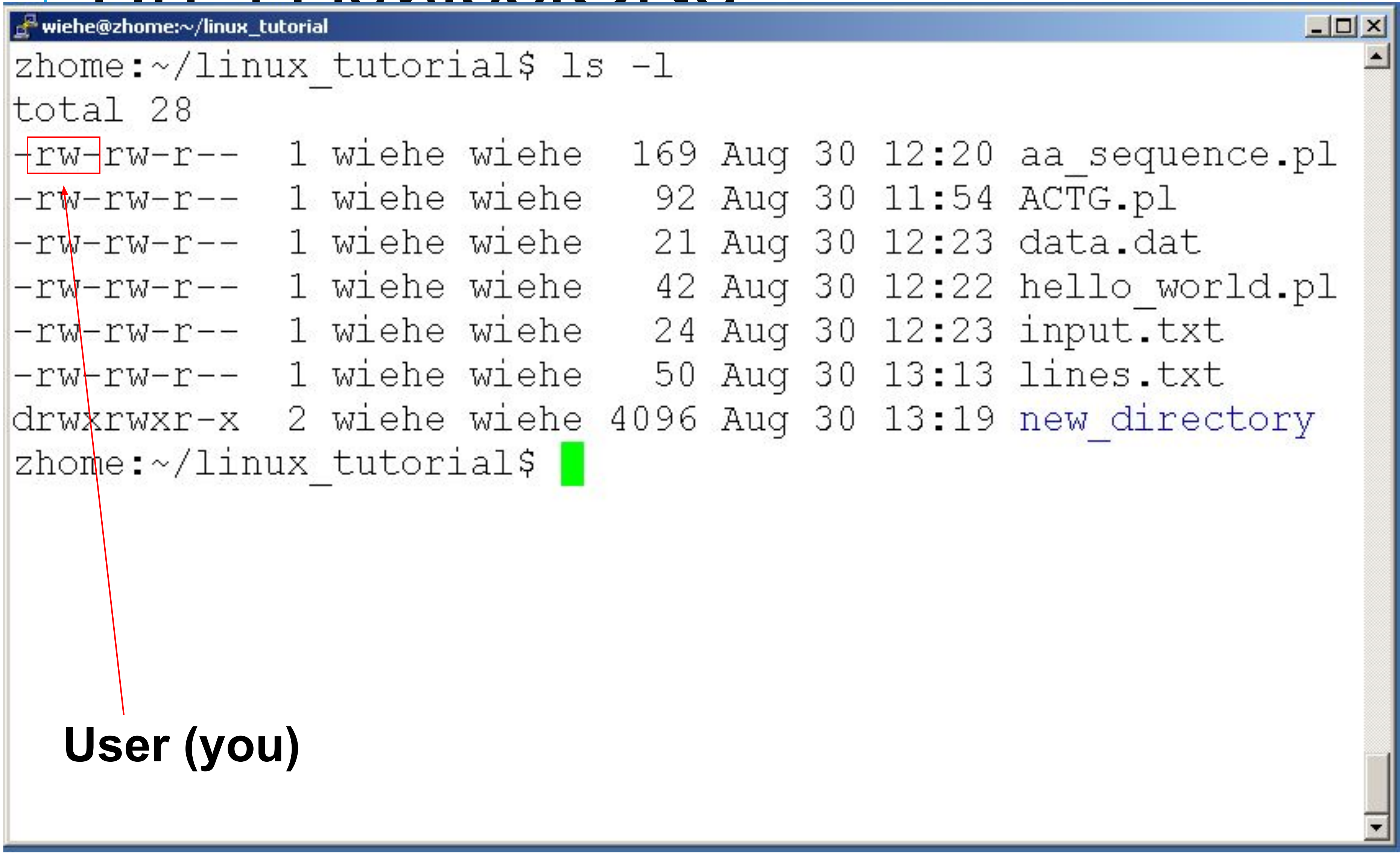
"r" means "read only" permission

"w" means "write" permission

"x" means "execute" permission

- In case of directory, "**x**" grants permission to list directory contents

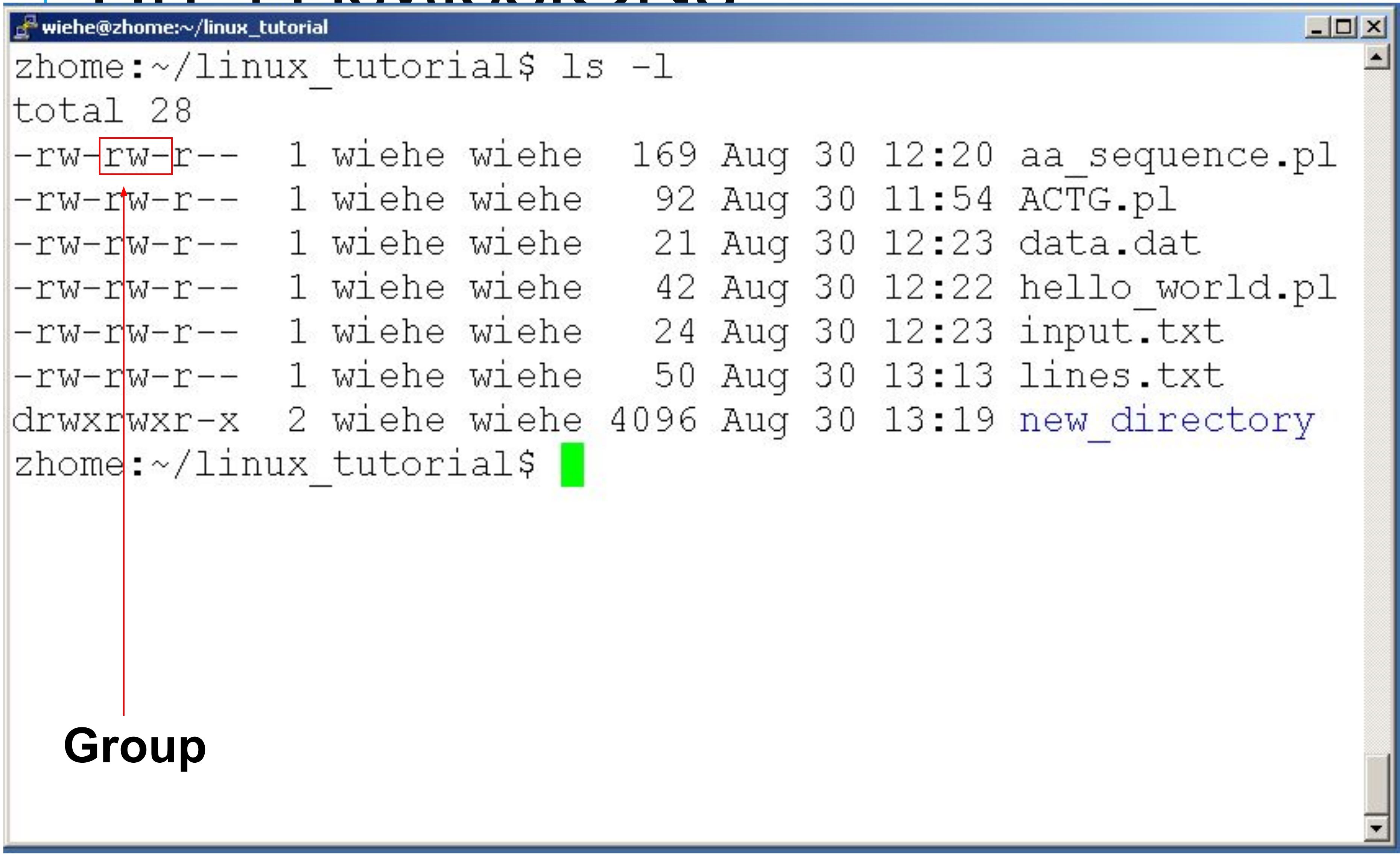
FILE PERMISSIONS



```
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)

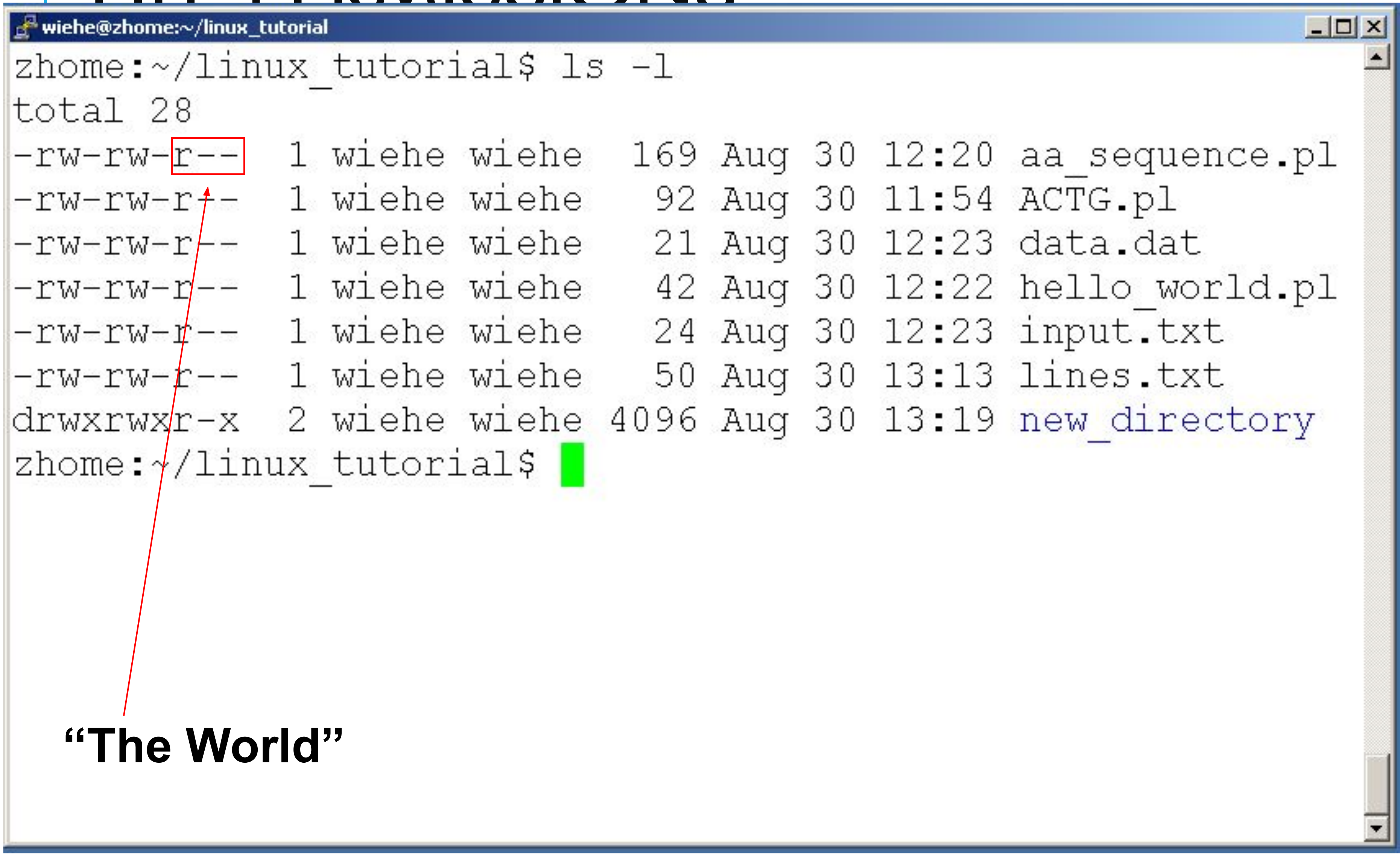
FILE PERMISSIONS


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



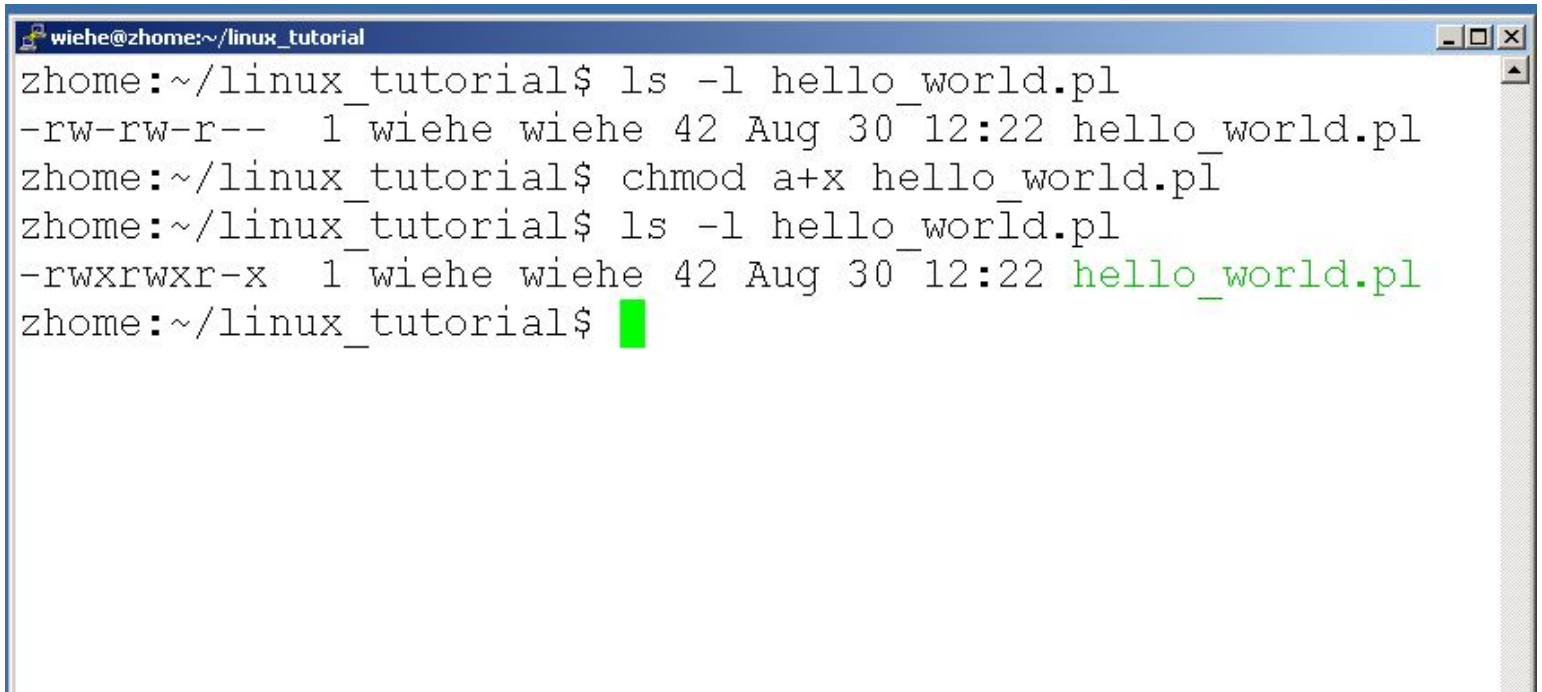
```
zhome:~/linux_tutorial$ ls -l
total 28
-rw-rw-r--  1 wiehe wiehe  169 Aug 30 12:20 aa_sequence.pl
-rw-rw-r--  1 wiehe wiehe   92 Aug 30 11:54 ACTG.pl
-rw-rw-r--  1 wiehe wiehe   21 Aug 30 12:23 data.dat
-rw-rw-r--  1 wiehe wiehe   42 Aug 30 12:22 hello_world.pl
-rw-rw-r--  1 wiehe wiehe   24 Aug 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 [**u**ser/**g**roup/**o**thers/**a**ll]+[permission] [file(s)]
- Below we grant execute permission to all:

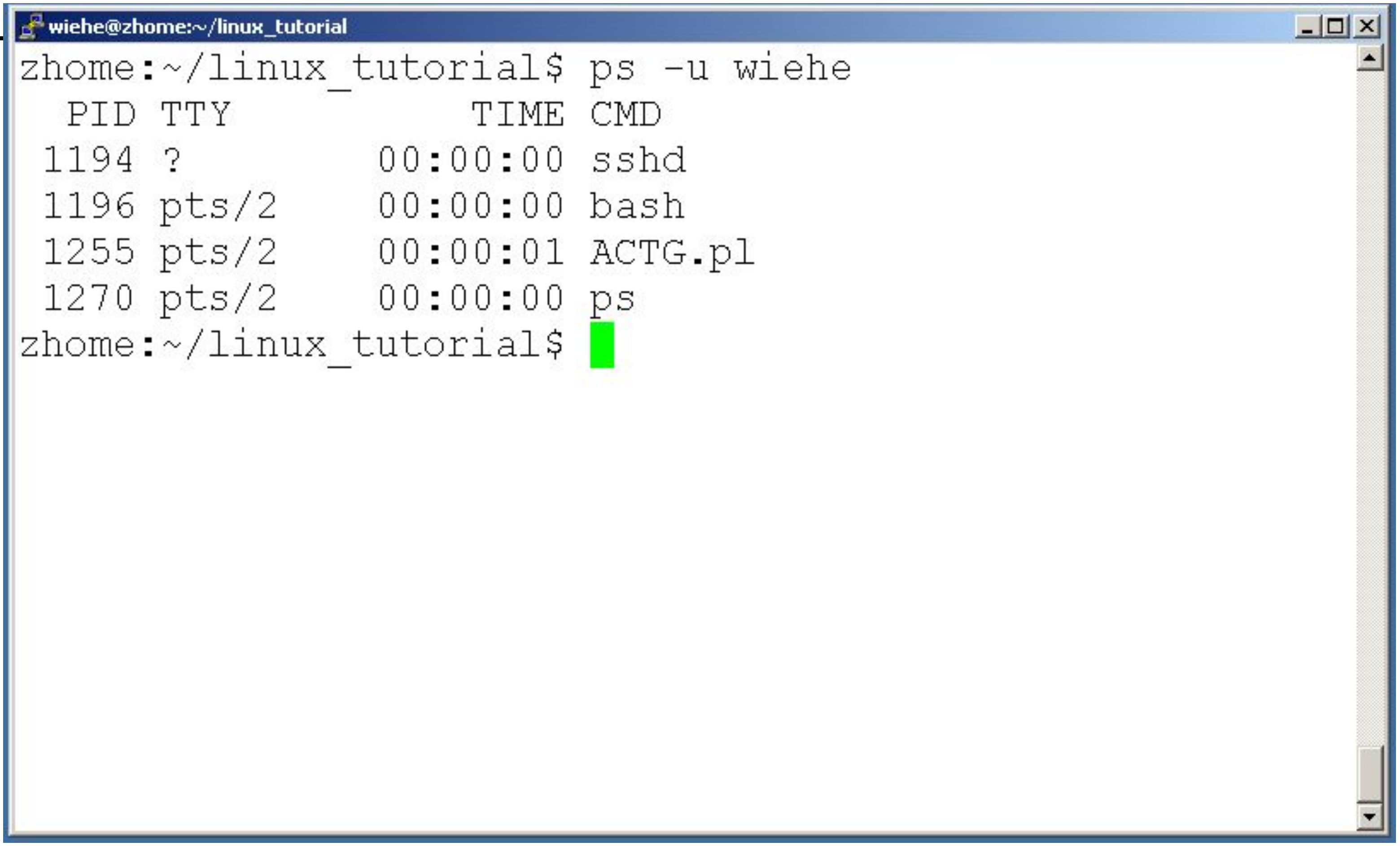


wiehe@zhome:~/linux_tutorial

```
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$
```

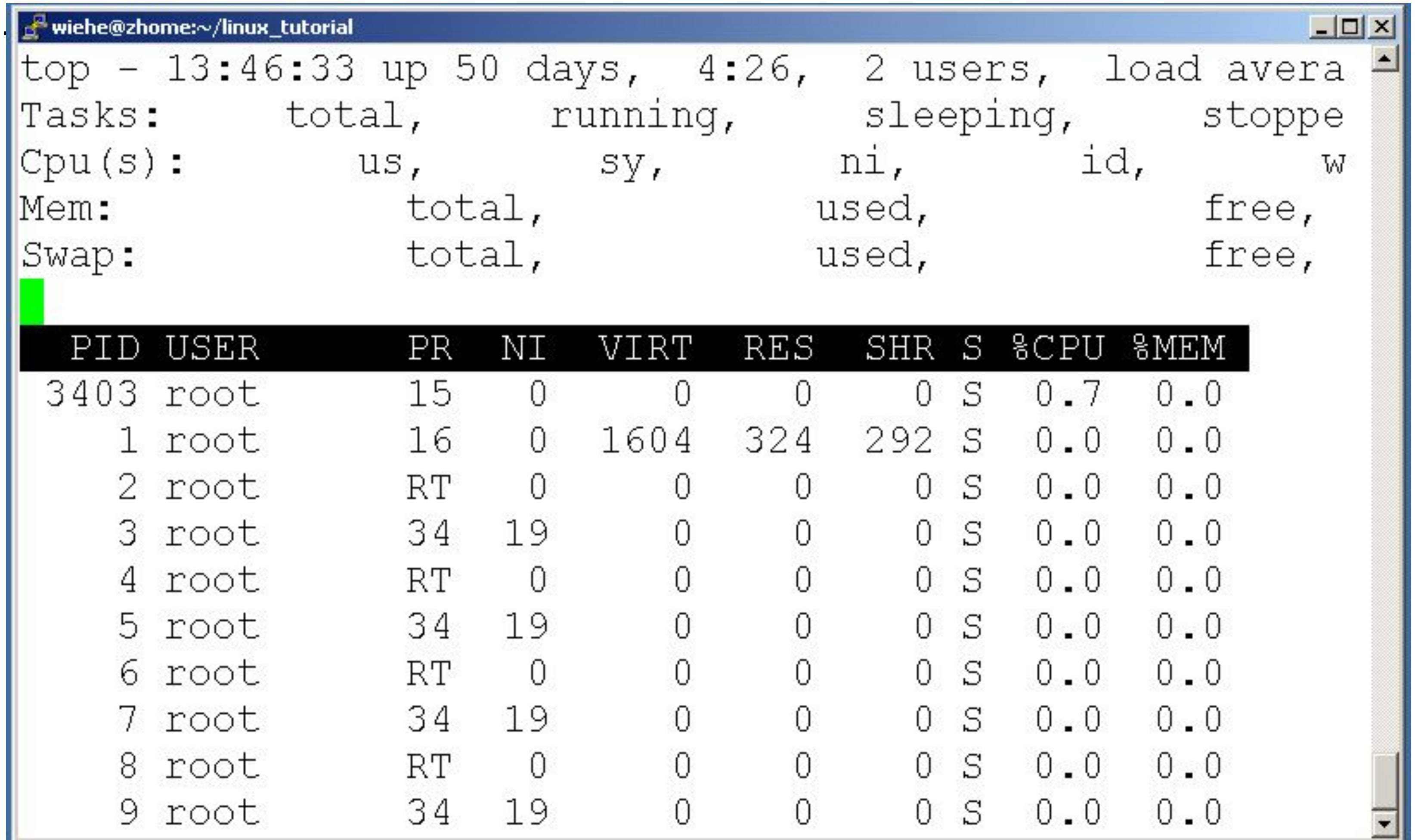
COMMAND: PS

T



```
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$
```

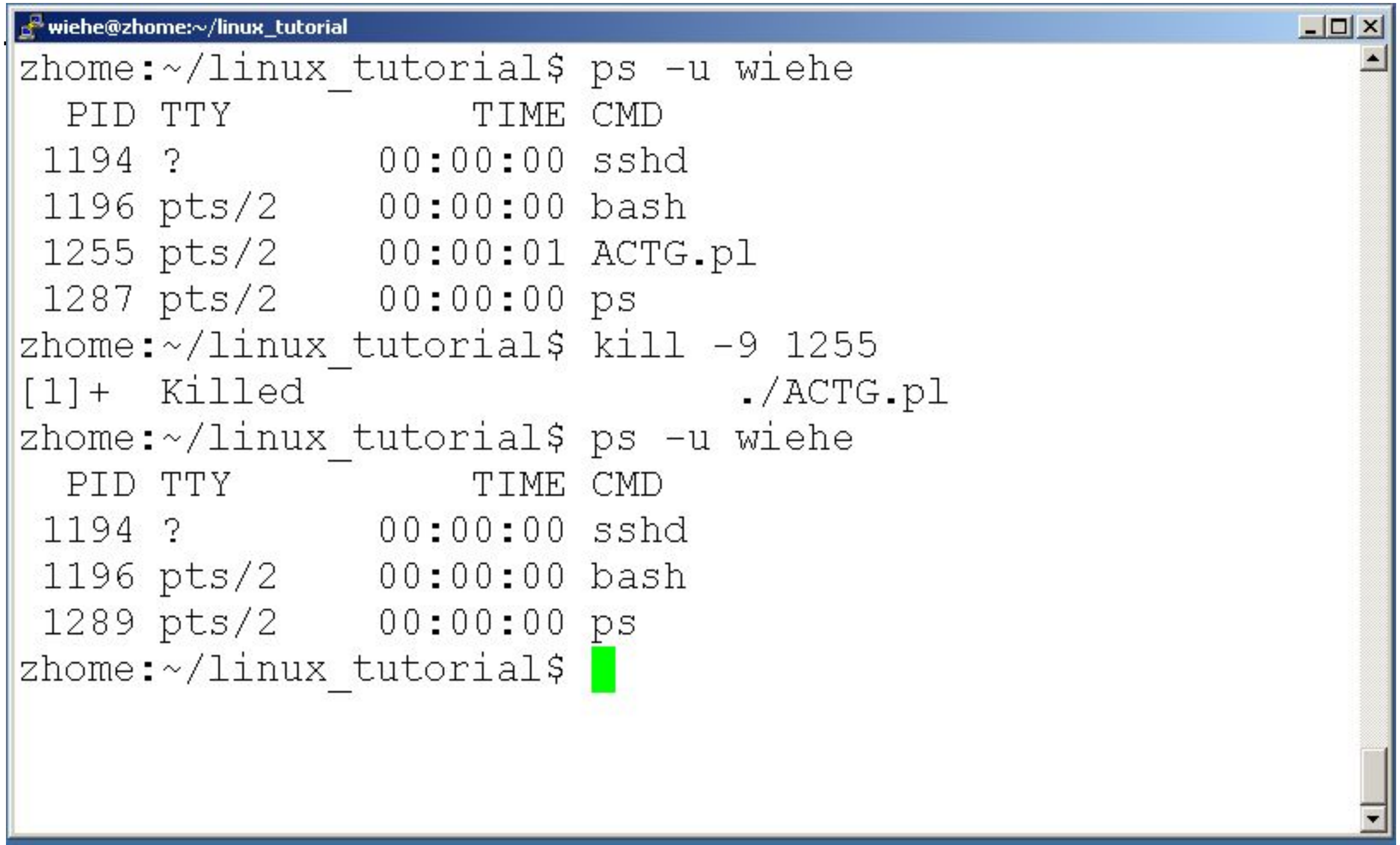
COMMAND: TOP



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



```
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

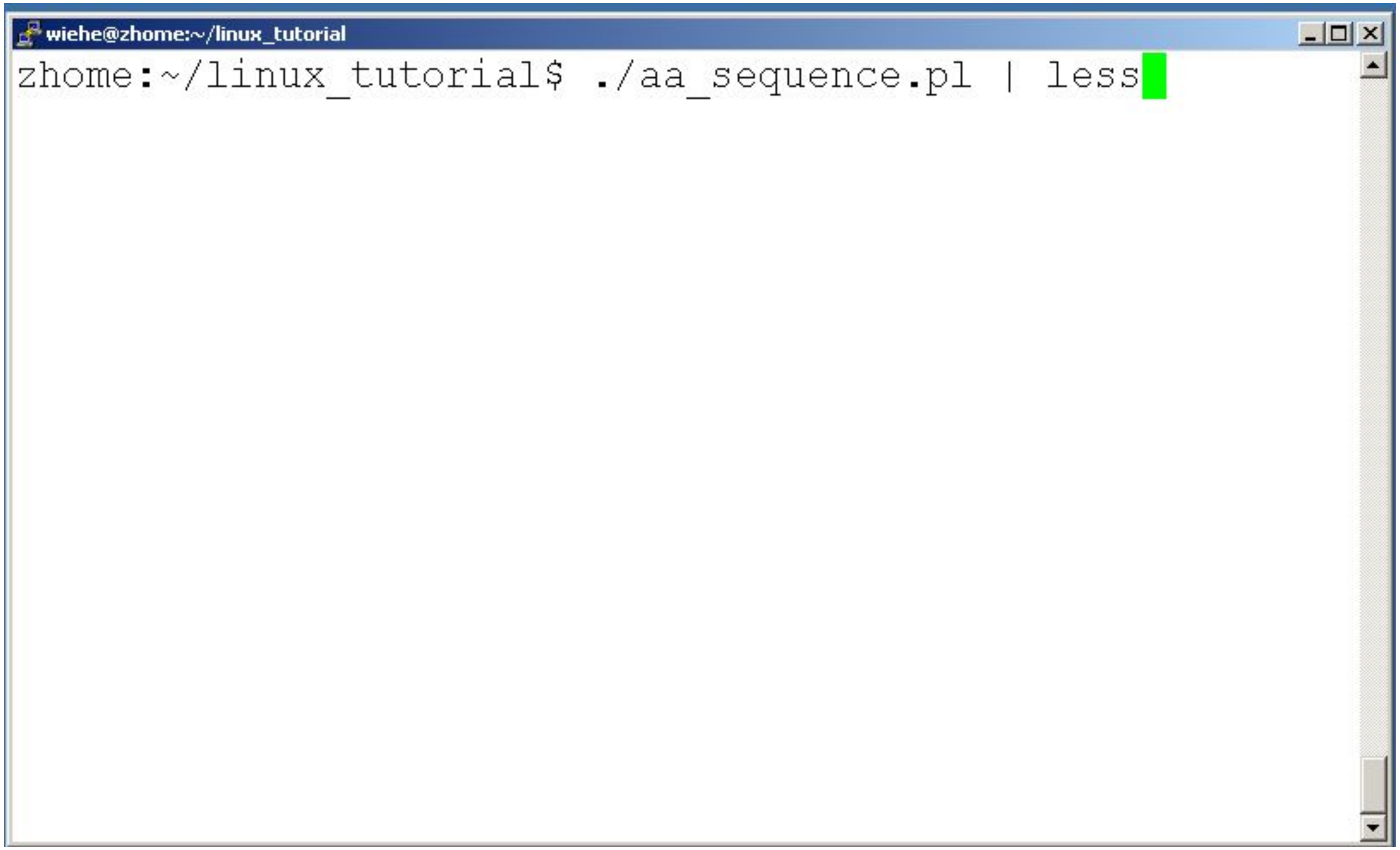
"program_a > file.txt"

- program_a's output is written to a file called "file.txt"

"program_a < input.txt"

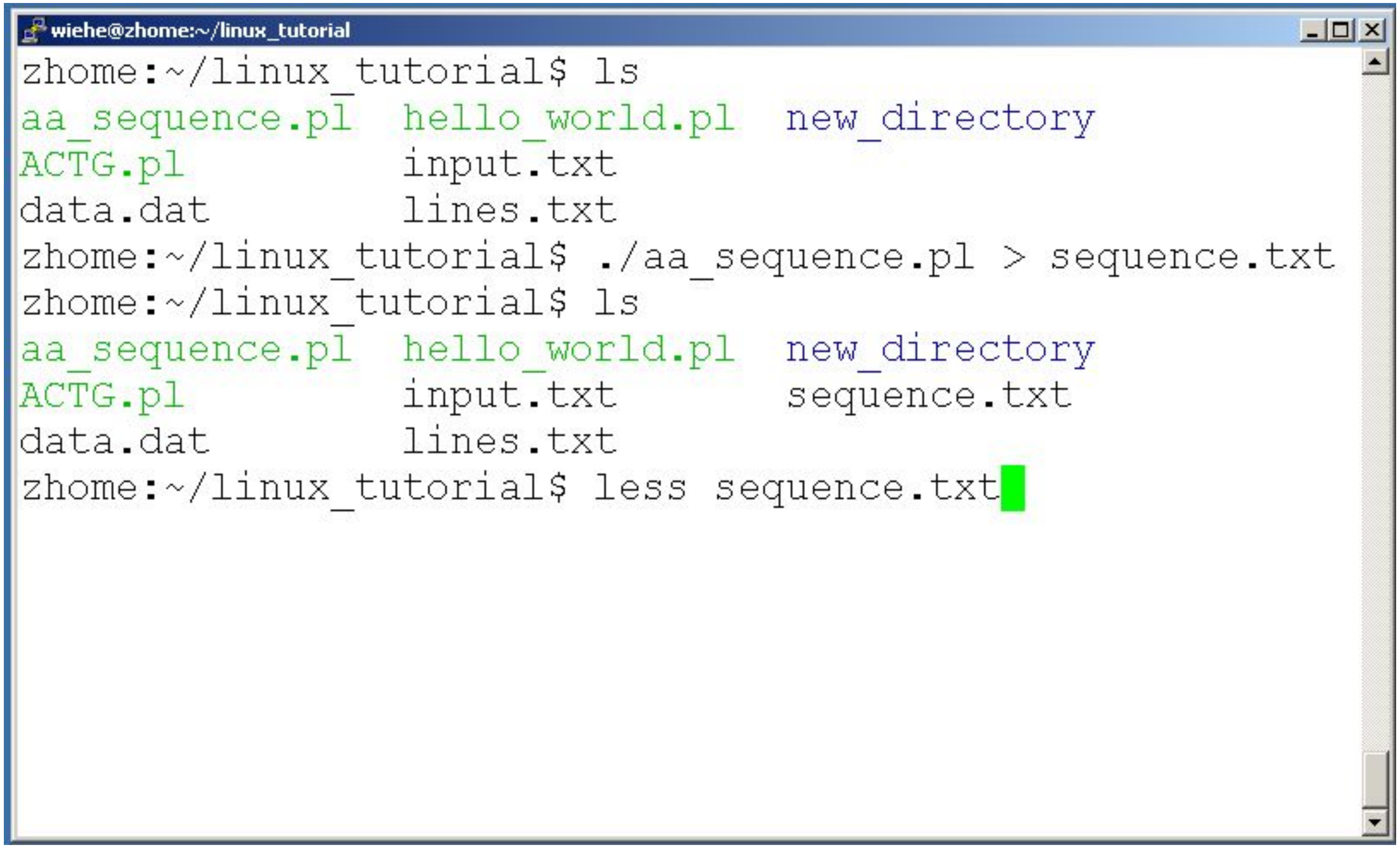
- program_a gets its input from a file called "input.txt"

A FEW EXAMPLES OF PIPING



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

A FEW EXAMPLES OF PIPING



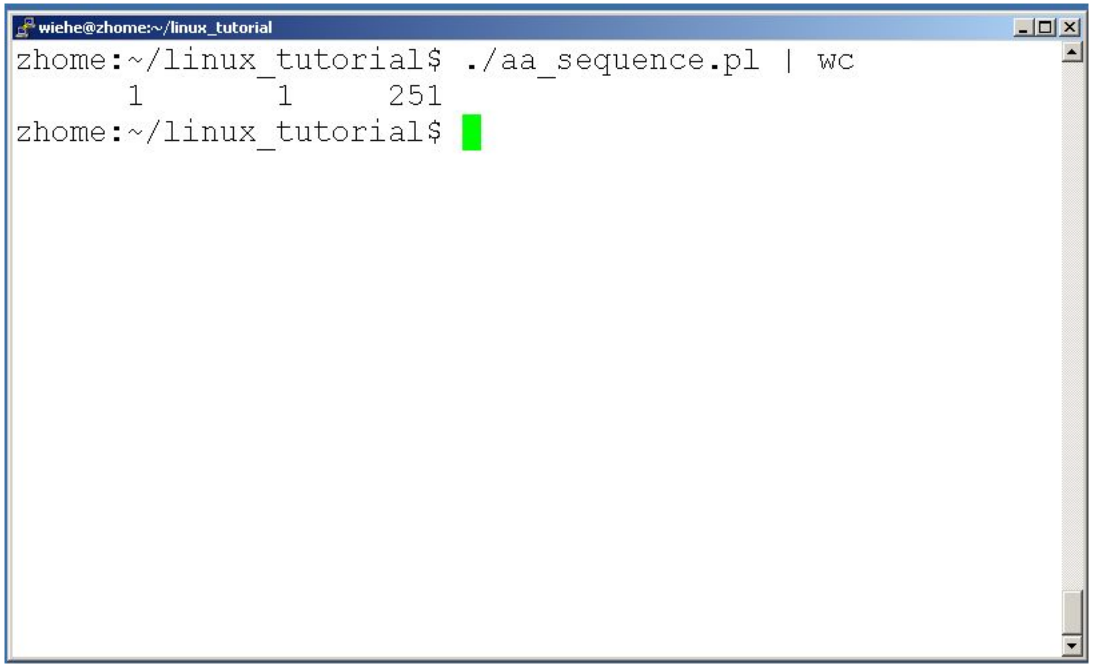
```
zhome:~/linux_tutorial$ ls
aa_sequence.pl   hello_world.pl   new_directory
ACTG.pl          input.txt
data.dat         lines.txt
zhome:~/linux_tutorial$ ./aa_sequence.pl > sequence.txt
zhome:~/linux_tutorial$ ls
aa_sequence.pl   hello_world.pl   new_directory
ACTG.pl          input.txt        sequence.txt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

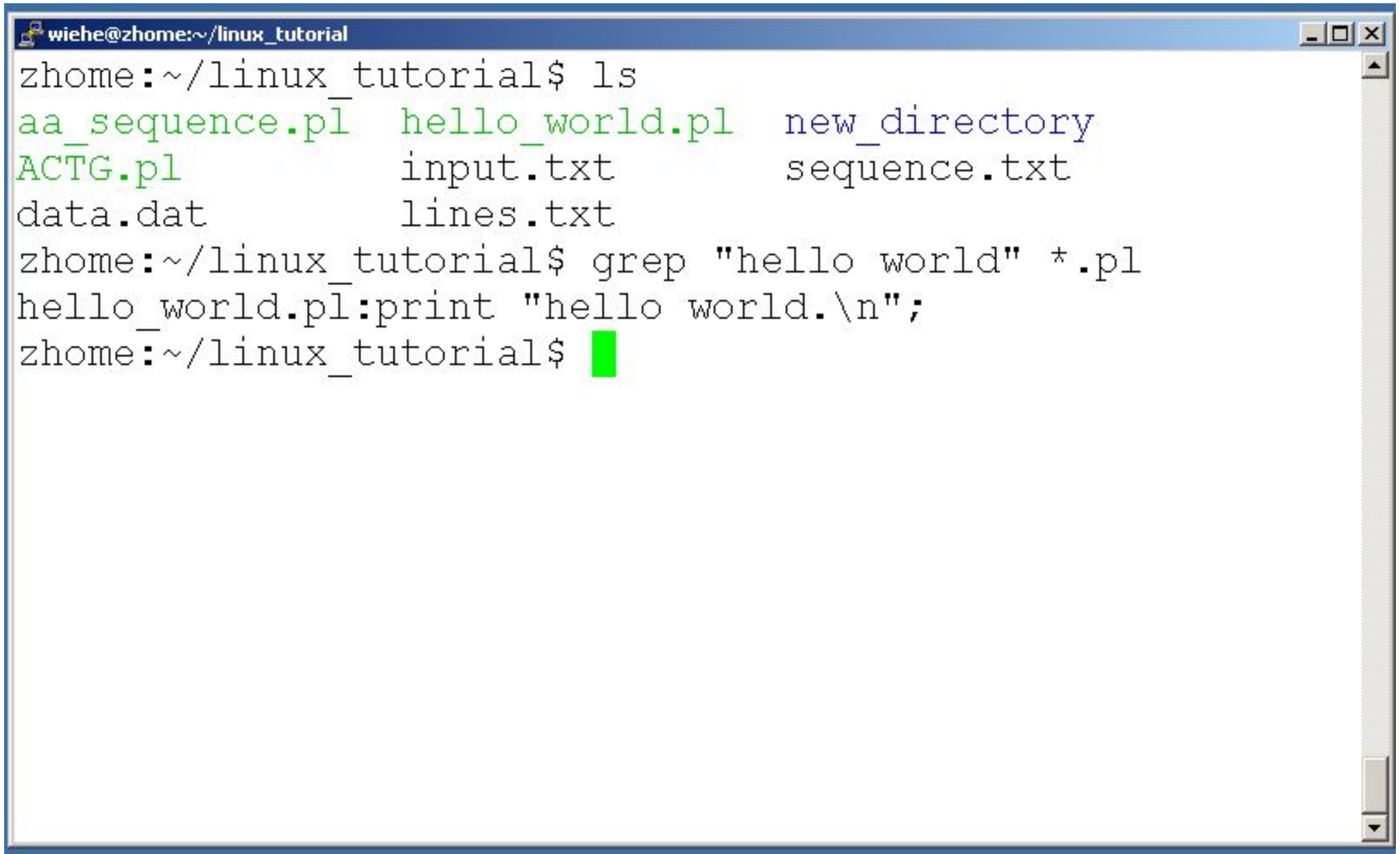
A FEW EXAMPLES OF PIPING



```
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”



```
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