

# DEAC102 – Introduction to Linux Systems Administration

## Lab #3.1

### *Navigating the Filesystem*

**Name:** Aldenir Flauzino

**Student No.:** CT1010383

**Introduction:** In this Lab you will learn how to navigate and manage files and directories.

You are required to complete the lab and record your answers. Rename the file using your first name and the lab number, e.g. **washington3.1.docx**, and submit it through Canvas to receive marks for this lab.

#### **Tasks:**

In this lab, you will perform the following tasks:

- Navigate home and system directories.
- List files and directories.

#### **Equipment Required:**

1. Device with Linux (UBUNTU)

#### **Step 1:**

In this lab you will explore the concepts of files and directories.

On a Linux OS, data is stored in *files* and files are stored in *directories*. You may be used to the term **folders** to describe directories.

Directories are actually files, too; the data that they hold are the names of the files that have been entered into them, along with the inode number (a unique identifier number assigned to each file) for where the data for that file exists on the disk.

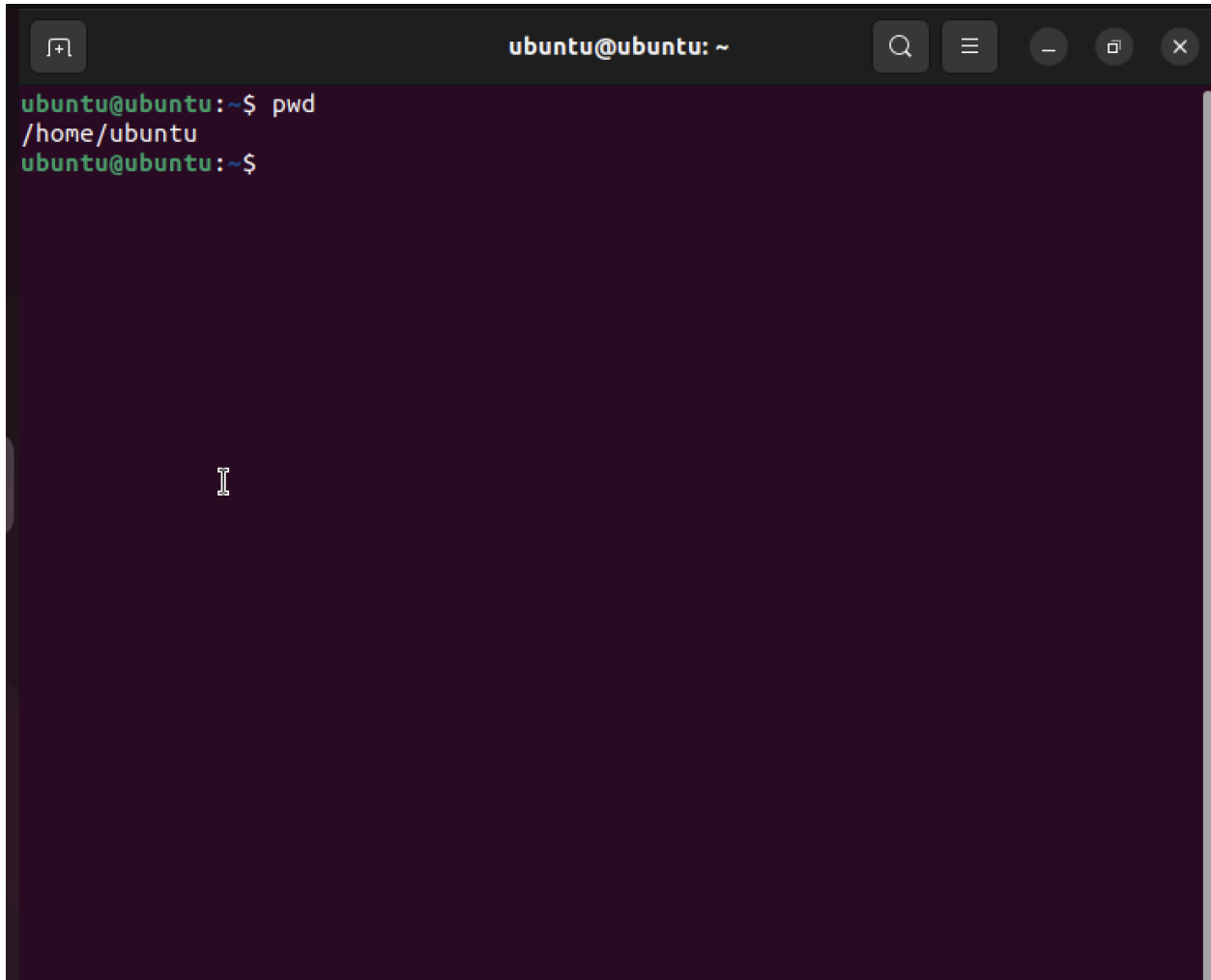
As a Linux user, you will want to know how to manipulate these files and directories, including how to list files in a directory, copy, delete and move files.

#### **Warning**

File and directory names in Linux are case-sensitive. This means that a file named **ABC** is not the same as a file named **abc**.

Type the following command to print the *working directory*:

```
pwd
```

A terminal window with a dark purple background. The title bar at the top reads 'ubuntu@ubuntu: ~'. On the left of the title bar is a window icon, and on the right are search, menu, and window control icons. The terminal content shows the command 'pwd' being entered at the prompt 'ubuntu@ubuntu:~\$'. The output is '/home/ubuntu', followed by a new prompt 'ubuntu@ubuntu:~\$'. A white cursor is visible on the line following the second prompt.

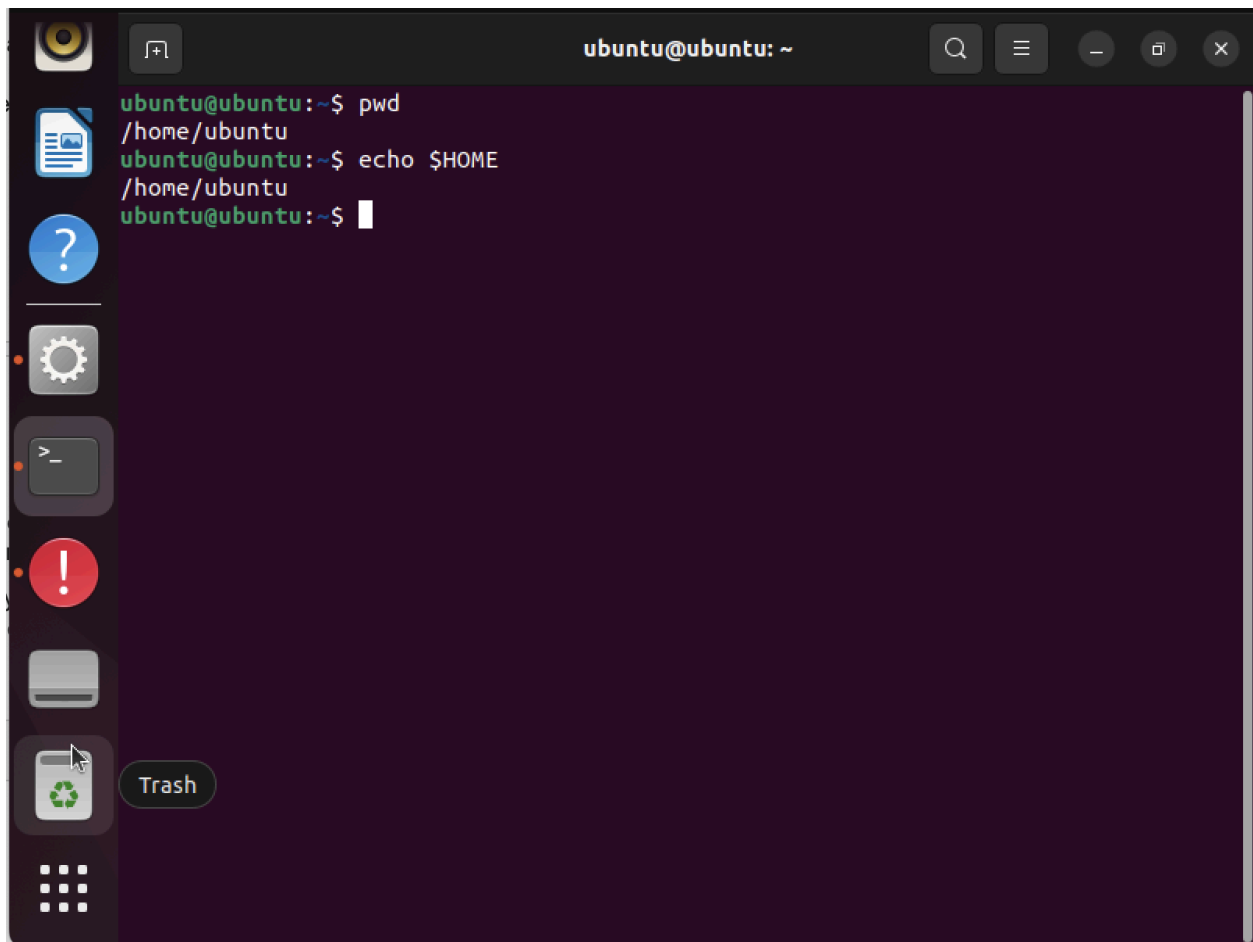
```
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$
```

The *working directory* is the directory that your terminal window is currently "in". This is also called the *current directory*. This will be important for when you are running subsequent commands, as they will behave differently based on the directory you are currently in.

The output of the `pwd` command is called the **path**. The first slash represents the **root directory**, the top level of the directory structure.

When you first open a terminal window, you will be placed in your **home directory**. This is a directory where you have full access and other users normally have no access by default. To see the path to your home directory, you can also execute the following command to view the value of the HOME variable:

```
echo $HOME
```

A screenshot of a terminal window titled 'ubuntu@ubuntu: ~'. The window has a dark purple background. On the left side, there is a vertical dock with several icons: a yellow circle with a black dot, a blue document icon, a blue circle with a white question mark, a grey gear icon, a grey terminal icon, a red circle with a white exclamation mark, a grey printer icon, a green recycling icon labeled 'Trash', and a 3x3 grid of white dots. The terminal text shows the following commands and output:

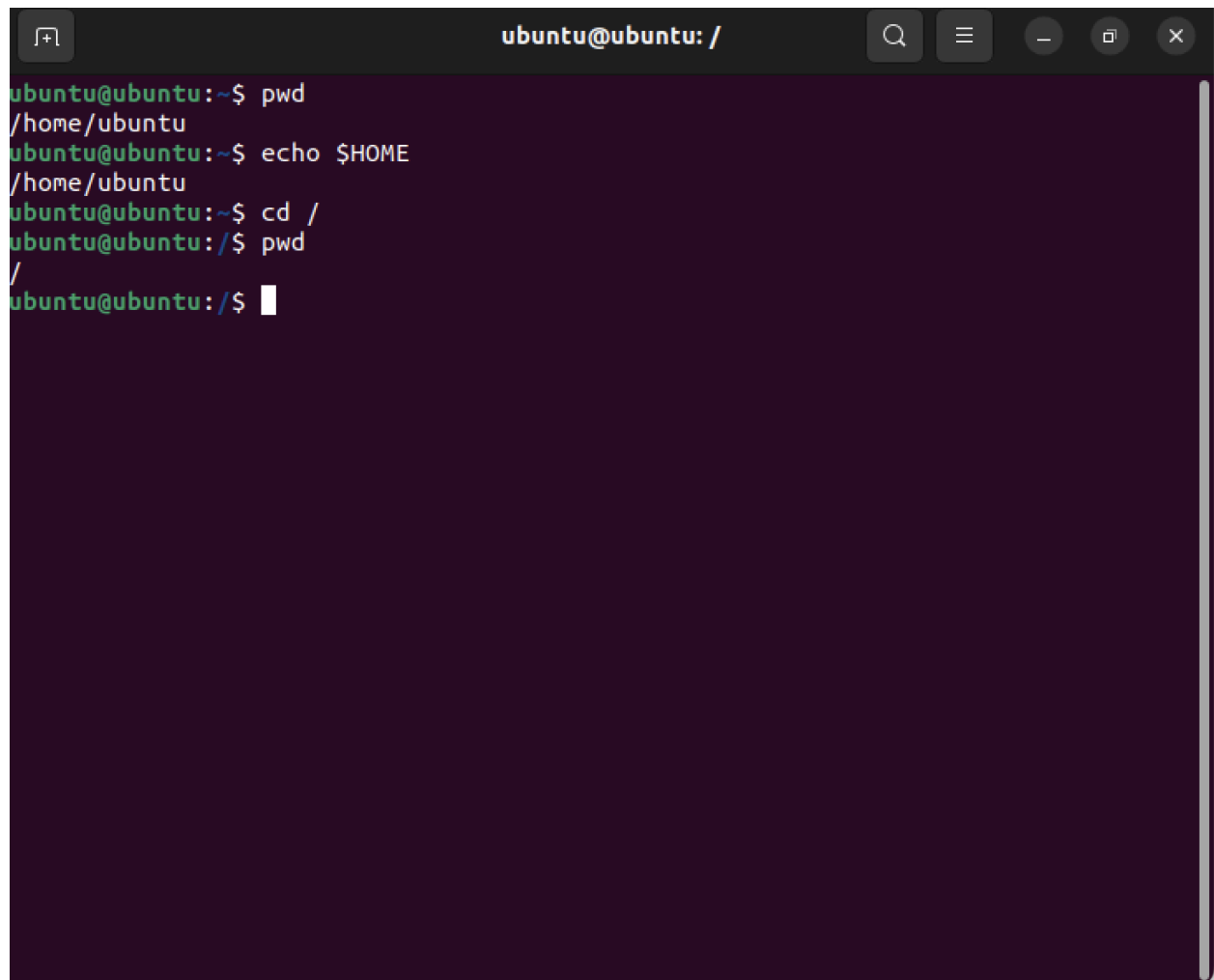
```
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo $HOME
/home/ubuntu
ubuntu@ubuntu:~$
```

## Step 2:

You can use the `cd` command with a path to a directory to change your current directory. Type the following command to make the root directory your current working directory and verify with the `pwd` command:

```
cd /
```

```
pwd
```



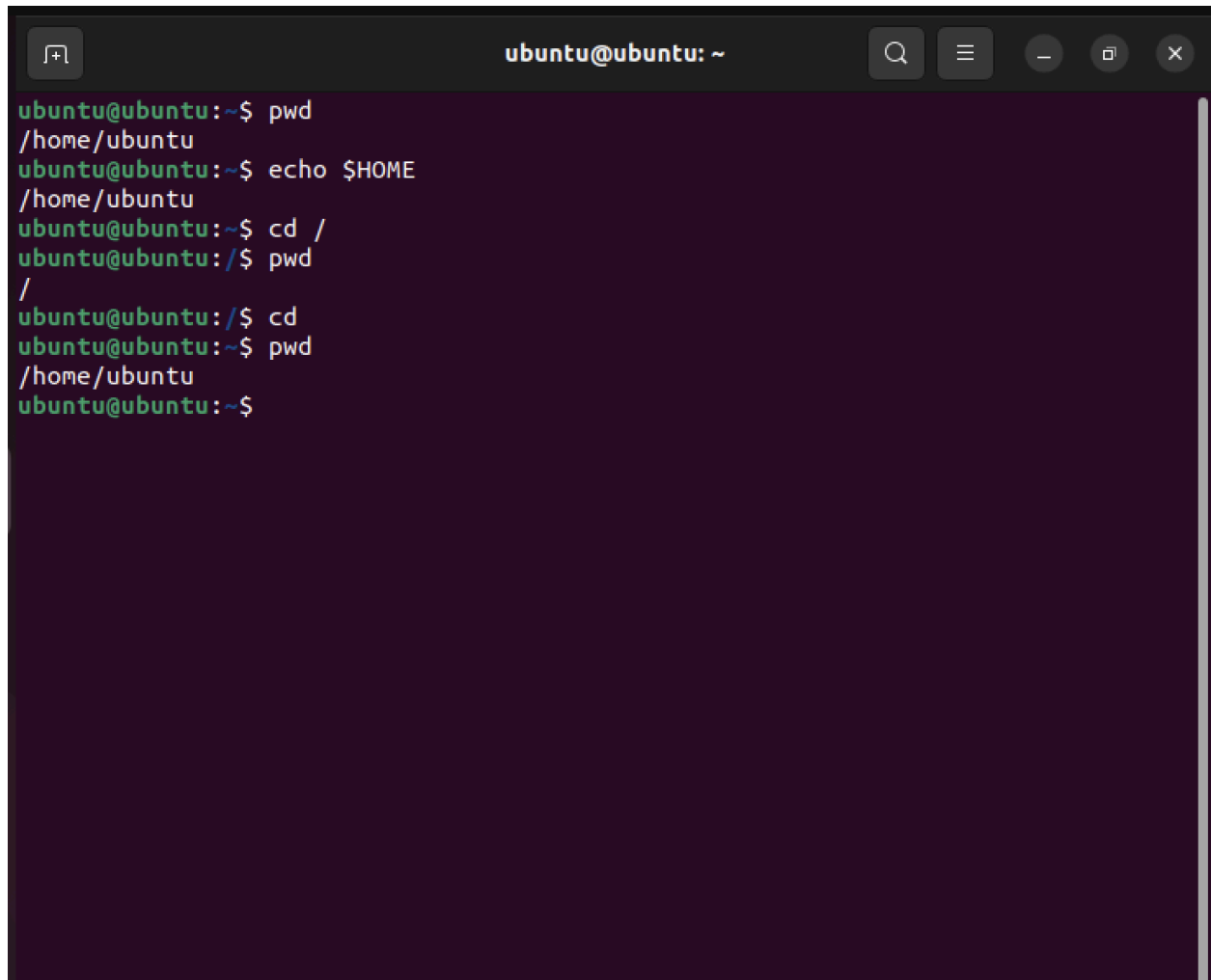
```
ubuntu@ubuntu: /  
ubuntu@ubuntu:~$ pwd  
/home/ubuntu  
ubuntu@ubuntu:~$ echo $HOME  
/home/ubuntu  
ubuntu@ubuntu:~$ cd /  
ubuntu@ubuntu:/$ pwd  
/  
ubuntu@ubuntu:/$
```

### Step 3:

To change back to your home directory, the `cd` command can be executed **without a path**. Change back to your home directory and verify by typing the following commands:

```
cd  
pwd
```

Notice the change in the *prompt*. The tilde `~` character represents your home directory. This part of the prompt will tell you what directory you are currently in.



```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ pwd  
/home/ubuntu  
ubuntu@ubuntu:~$ echo $HOME  
/home/ubuntu  
ubuntu@ubuntu:~$ cd /  
ubuntu@ubuntu:/$ pwd  
/  
ubuntu@ubuntu:/$ cd  
ubuntu@ubuntu:~$ pwd  
/home/ubuntu  
ubuntu@ubuntu:~$
```

#### Step 4:

The `cd` command may be entered with a path to a directory specified as an *argument*. Execute the `cd` command with the **/home** directory as an argument by typing the following:

```
cd /home  
pwd
```

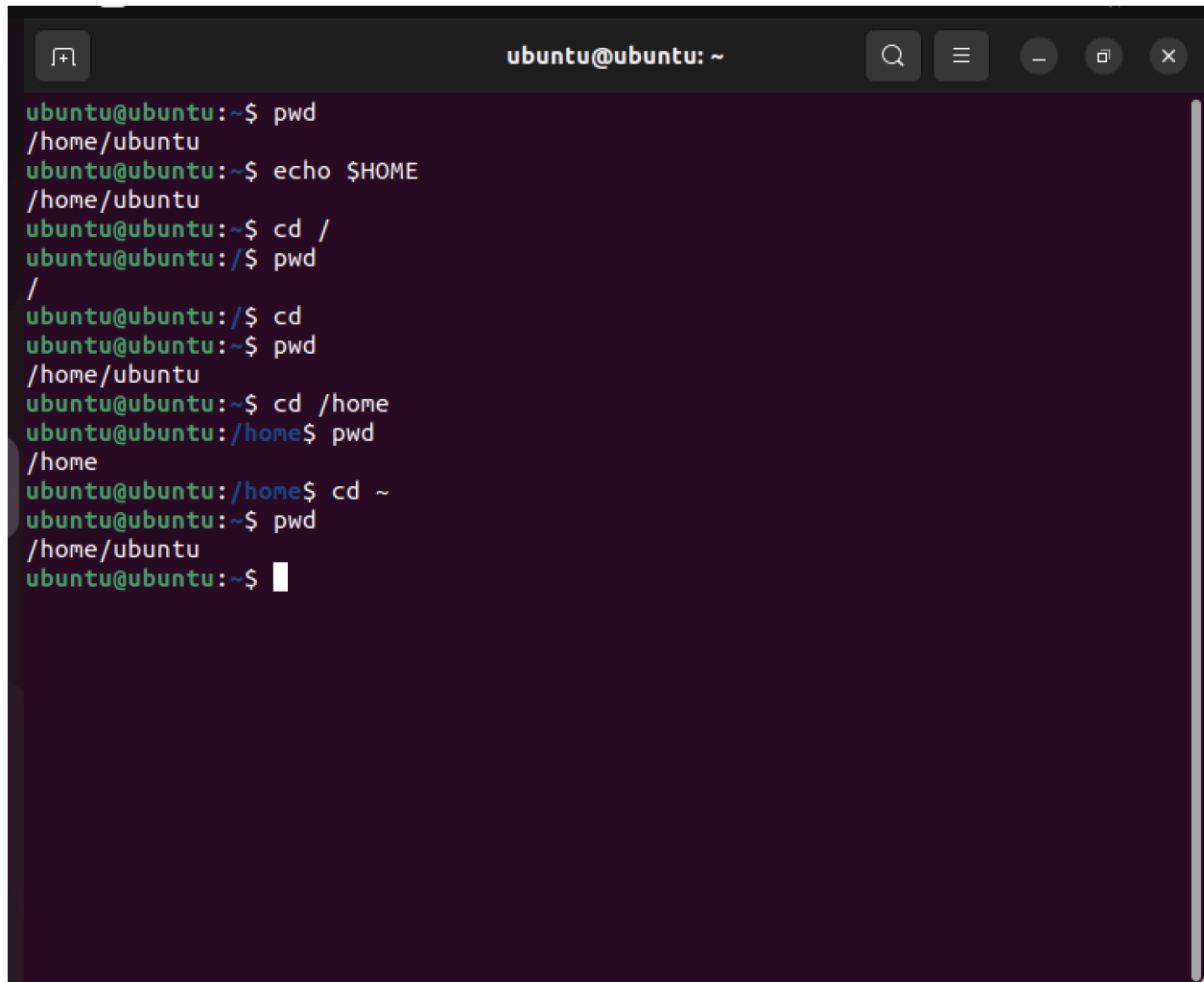
```
ubuntu@ubuntu: /home
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo $HOME
/home/ubuntu
ubuntu@ubuntu:~$ cd /
ubuntu@ubuntu:/$ pwd
/
ubuntu@ubuntu:/$ cd
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ cd /home
ubuntu@ubuntu:/home$ pwd
/home
ubuntu@ubuntu:/home$
```

When the path that is provided as an argument to the `cd` command starts with the forward slash `/`, that path is referred to as an “**absolute path**”. **Absolute paths are always complete paths from the root directory to a subdirectory or file.**

### Step 5:

Change back to your home directory, using the `cd` command with the tilde `~` as an argument:

```
cd ~
pwd
```

A terminal window titled 'ubuntu@ubuntu: ~' with standard window controls. The terminal shows a series of commands and their outputs: 'pwd' returns '/home/ubuntu', 'echo \$HOME' returns '/home/ubuntu', 'cd /' changes the directory to root, 'pwd' returns '/', 'cd' changes back to the home directory, 'pwd' returns '/home/ubuntu', 'cd /home' changes to the home directory, 'pwd' returns '/home', 'cd ~' changes back to the home directory, and 'pwd' returns '/home/ubuntu'. The prompt is now 'ubuntu@ubuntu:~\$' with a cursor.

```
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo $HOME
/home/ubuntu
ubuntu@ubuntu:~$ cd /
ubuntu@ubuntu:/$ pwd
/
ubuntu@ubuntu:/$ cd
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ cd /home
ubuntu@ubuntu:/home$ pwd
/home
ubuntu@ubuntu:/home$ cd ~
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$
```

When the path that is provided as an argument to the `cd` command starts with a **tilde ~ character**, the terminal will expand the character to the home directory of a user with an account on the system.

If either no other characters or a forward slash follows the tilde, then it will expand to the home directory of **the user currently active** in the shell.

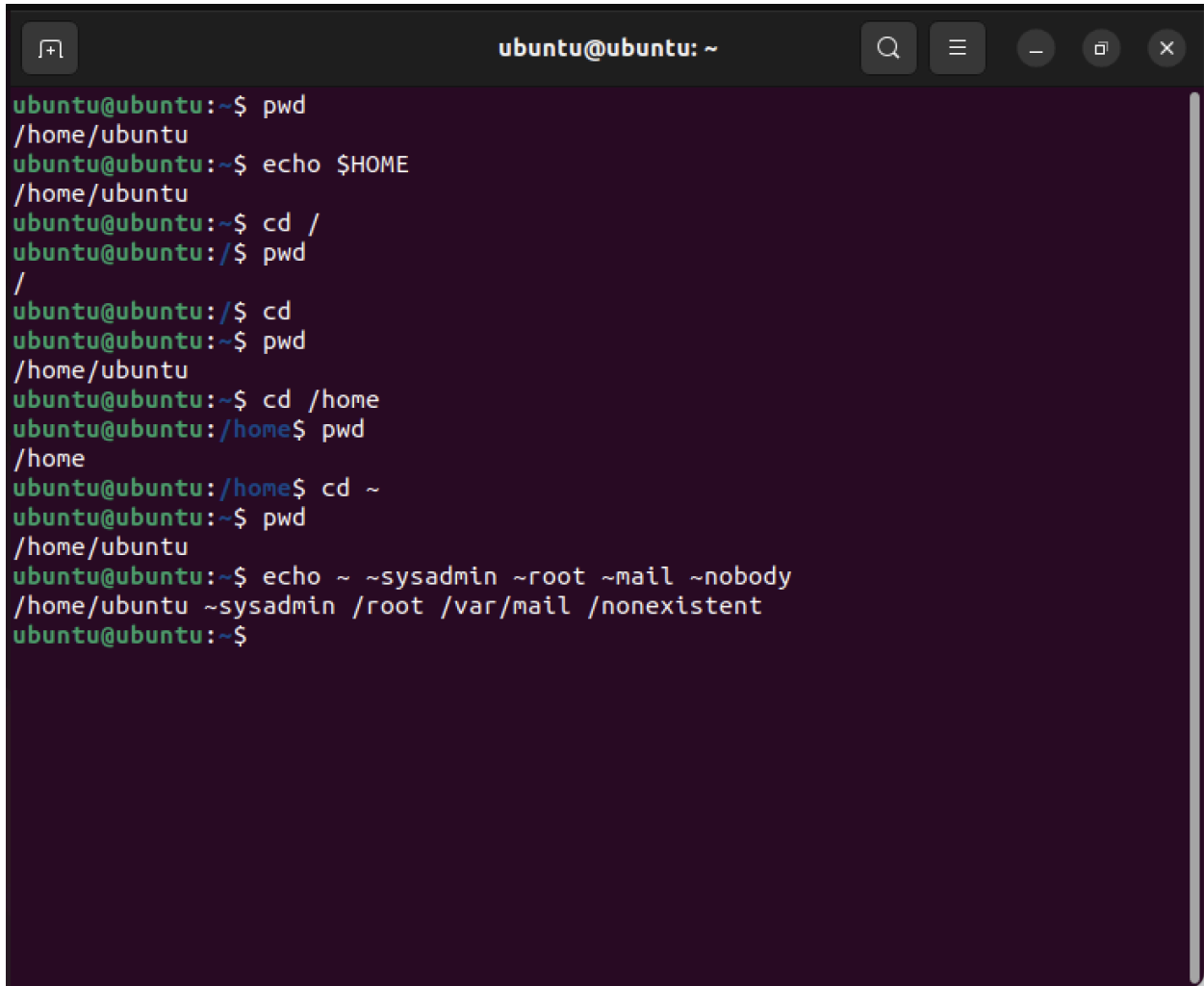
If a user name immediately follows the tilde character, then the shell will expand the tilde and user name to the home directory of that user name. **For example, ~bob would be expanded to /home/bob.**

**Paths that start with a tilde are considered absolute paths because after the shell expands the tilde path, an absolute path is formed.**

**Step 6:**

Use the `echo` command below to display some other examples of using the tilde as part of the path:

```
echo ~ ~sysadmin ~root ~mail ~nobody
```

A terminal window titled 'ubuntu@ubuntu: ~' with standard window controls. It shows a series of commands and their outputs: 'pwd' returns '/home/ubuntu'; 'echo \$HOME' returns '/home/ubuntu'; 'cd /' followed by 'pwd' returns '/'; 'cd' followed by 'pwd' returns '/home/ubuntu'; 'cd /home' followed by 'pwd' returns '/home'; 'cd ~' followed by 'pwd' returns '/home/ubuntu'. The final command is 'echo ~ ~sysadmin ~root ~mail ~nobody', which outputs the paths for each user: '/home/ubuntu ~sysadmin /root /var/mail /nonexistent'.

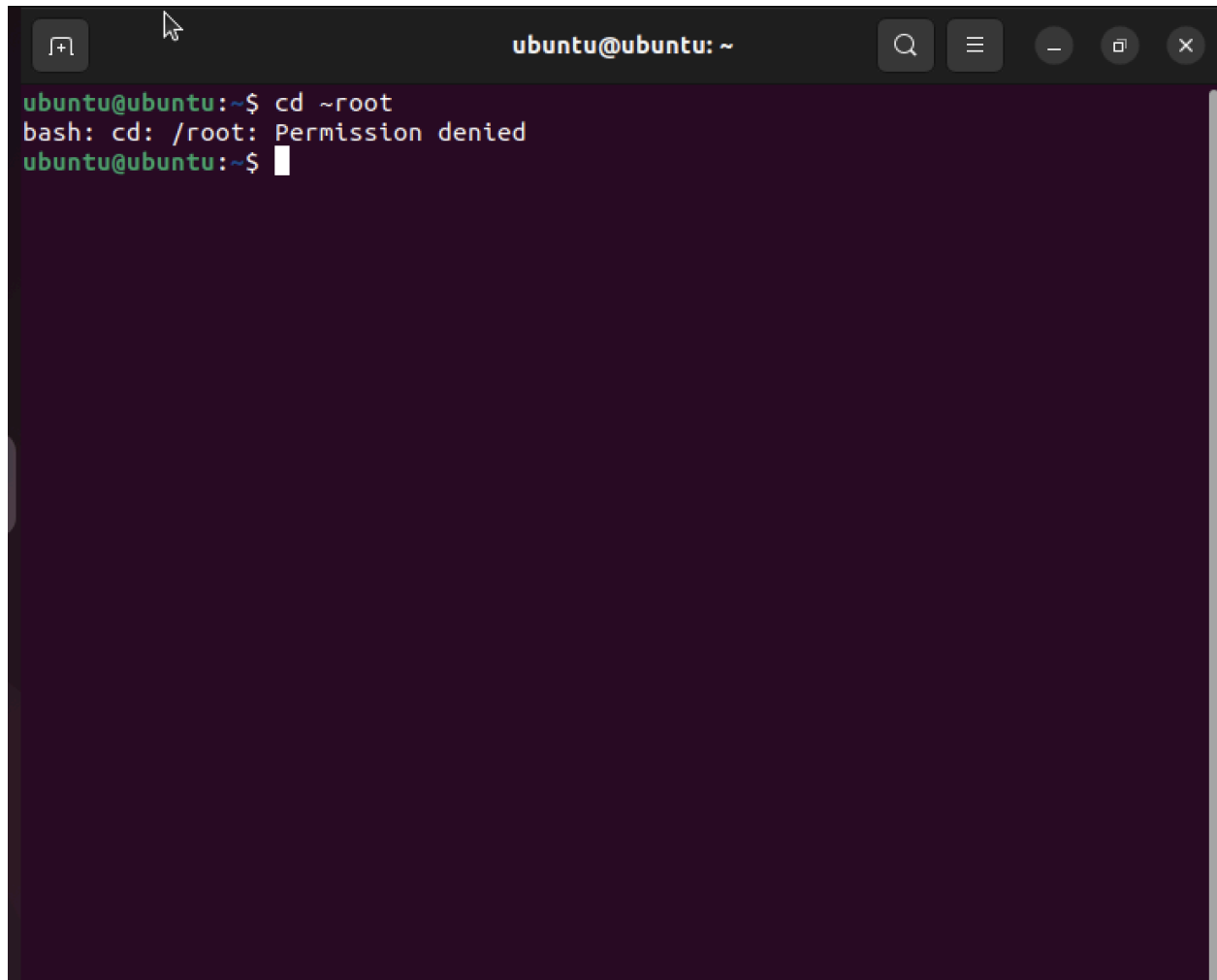
```
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo $HOME
/home/ubuntu
ubuntu@ubuntu:~$ cd /
ubuntu@ubuntu:/$ pwd
/
ubuntu@ubuntu:/$ cd
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ cd /home
ubuntu@ubuntu:/home$ pwd
/home
ubuntu@ubuntu:/home$ cd ~
ubuntu@ubuntu:~$ pwd
/home/ubuntu
ubuntu@ubuntu:~$ echo ~ ~sysadmin ~root ~mail ~nobody
/home/ubuntu ~sysadmin /root /var/mail /nonexistent
ubuntu@ubuntu:~$
```

### Step 7:

Attempt to change to the home directory of the root user by typing the following command:

```
cd ~root
```



A terminal window with a dark background. The title bar shows 'ubuntu@ubuntu: ~'. The prompt is 'ubuntu@ubuntu:~\$'. The user enters 'cd ~root'. The terminal displays the error message 'bash: cd: /root: Permission denied'. The prompt returns to 'ubuntu@ubuntu:~\$' with a cursor.

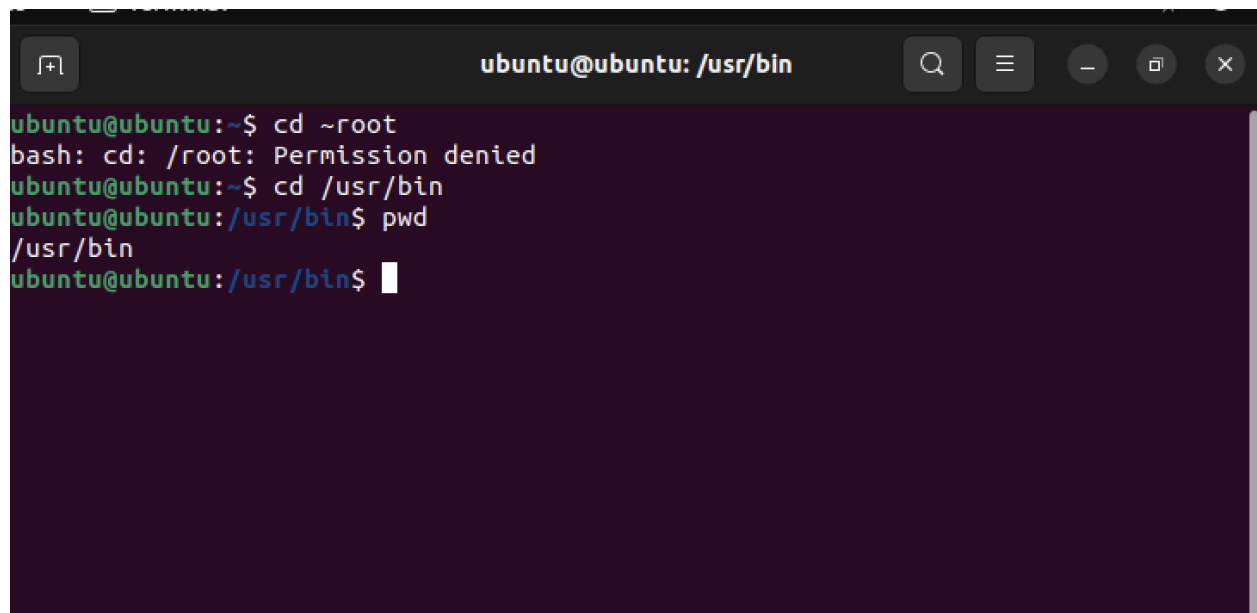
```
ubuntu@ubuntu:~$ cd ~root
bash: cd: /root: Permission denied
ubuntu@ubuntu:~$
```

Notice the error message; it indicates that the shell attempted to execute `cd` with `/root` as an argument but it failed due to permission being denied.

#### Step 8:

Using an **absolute path**, change to the `/usr/bin` directory and display the working directory by using the following commands:

```
cd /usr/bin
pwd
```

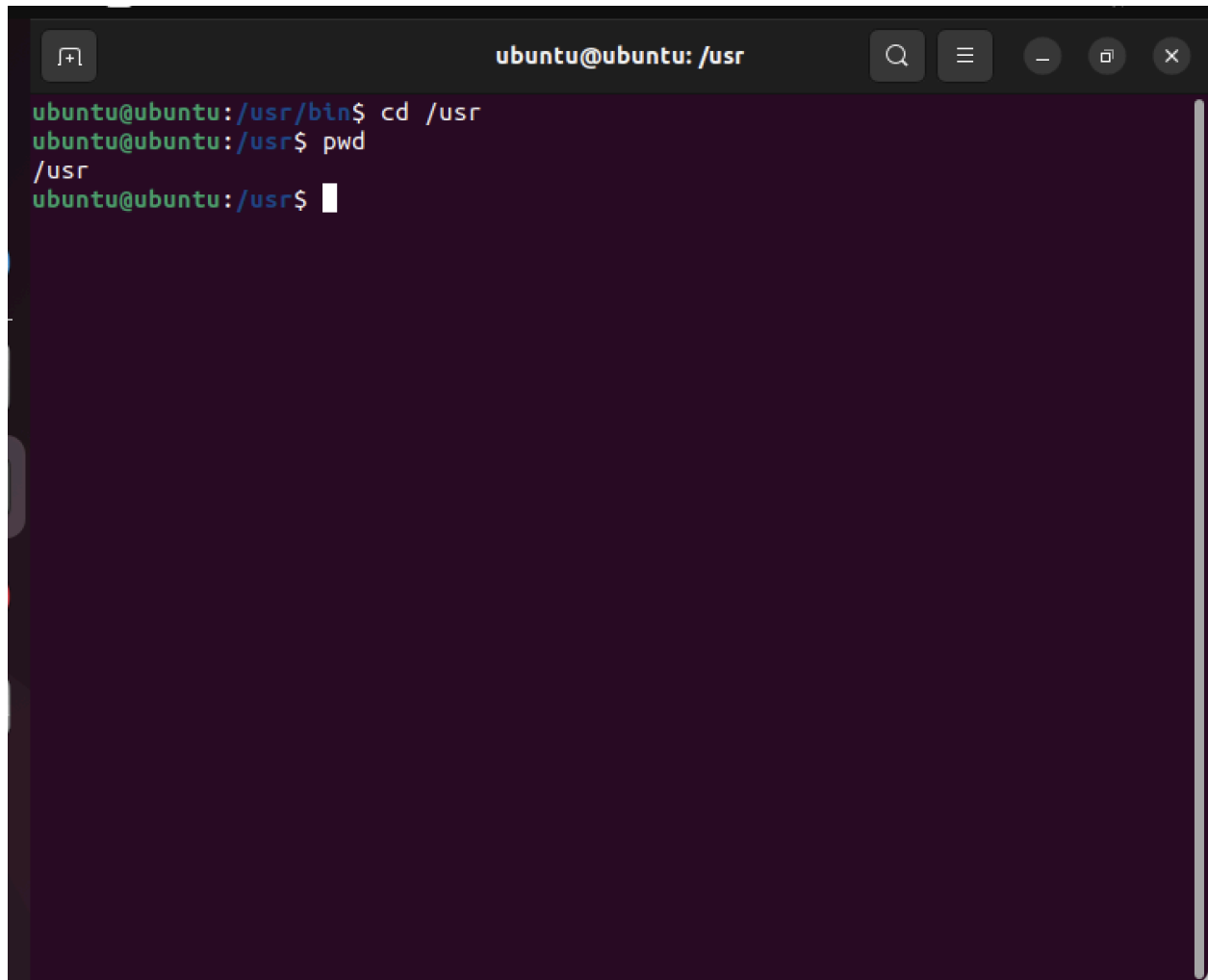
A terminal window with a dark background and light-colored text. The window title bar shows 'ubuntu@ubuntu: /usr/bin'. The terminal content shows a sequence of commands and their outputs: 'cd ~root' results in 'bash: cd: /root: Permission denied'; 'cd /usr/bin' changes the directory; 'pwd' outputs '/usr/bin'.

```
ubuntu@ubuntu:~$ cd ~root
bash: cd: /root: Permission denied
ubuntu@ubuntu:~$ cd /usr/bin
ubuntu@ubuntu:/usr/bin$ pwd
/usr/bin
ubuntu@ubuntu:/usr/bin$
```

### Step 9:

Use an absolute path to change to the **/usr** directory and display the working directory by issuing the following commands:

```
cd /usr
pwd
```

A terminal window with a dark background and light-colored text. The window title is 'ubuntu@ubuntu: /usr'. The terminal shows the following commands and output:

```
ubuntu@ubuntu:/usr/bin$ cd /usr
ubuntu@ubuntu:/usr$ pwd
/usr
ubuntu@ubuntu:/usr$
```

### Step 10:

Use an absolute path to change to the **/usr/share/doc** directory and display the working directory by issuing the following commands:

```
cd /usr/share/doc
pwd
```

```
ubuntu@ubuntu:/usr/bin$ cd /usr
ubuntu@ubuntu:/usr$ pwd
/usr
ubuntu@ubuntu:/usr$ cd /usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$
```

share

### Absolute vs. Relative pathnames

Suppose you are in the `/usr/share/doc` directory, and you want to go to the `/usr/share/doc/bash` directory. Typing the command `cd /usr/share/doc/bash` results in a fair amount of typing. **In cases like this, you want to use relative pathnames.**

With relative pathnames you provide "**directions**" of where you want to go from the current directory.

### Step 11:

Using a relative path, change to the `/usr/share/doc/bash` directory and display the working directory by issuing the following commands:

```
cd bash
pwd
```

```
ubuntu@ubuntu: /usr/share/doc/bash
ubuntu@ubuntu:/usr/bin$ cd /usr
ubuntu@ubuntu:/usr$ pwd
/usr
ubuntu@ubuntu:/usr$ cd /usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ cd bash
ubuntu@ubuntu:/usr/share/doc/bash$ pwd
/usr/share/doc/bash
ubuntu@ubuntu:/usr/share/doc/bash$
```

### Note

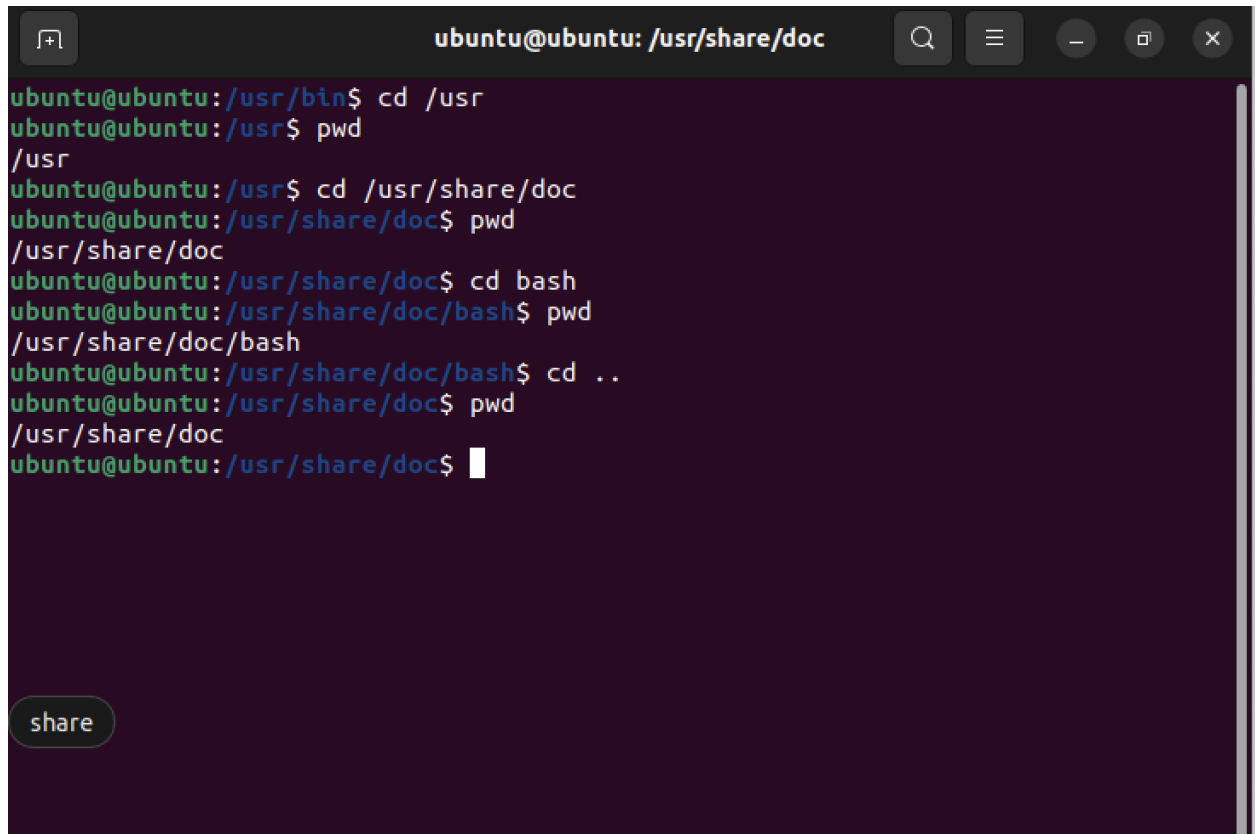
If there wasn't a bash directory under the current directory, the previous command would fail.

### Step 12:

Use a relative path to change to the directory above the current directory:

```
cd ..
pwd
```

The .. represents one level above your current directory location.

A terminal window titled 'ubuntu@ubuntu: /usr/share/doc' with standard window controls. The terminal shows a sequence of commands: 'cd /usr', 'pwd' (output: /usr), 'cd /usr/share/doc', 'pwd' (output: /usr/share/doc), 'cd bash', 'pwd' (output: /usr/share/doc/bash), 'cd ..', 'pwd' (output: /usr/share/doc), and finally 'cd' (output: /usr/share/doc). A 'share' button is visible in the bottom-left corner of the terminal area.

```
ubuntu@ubuntu: /usr/share/doc
ubuntu@ubuntu: /usr/bin$ cd /usr
ubuntu@ubuntu: /usr$ pwd
/usr
ubuntu@ubuntu: /usr$ cd /usr/share/doc
ubuntu@ubuntu: /usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu: /usr/share/doc$ cd bash
ubuntu@ubuntu: /usr/share/doc/bash$ pwd
/usr/share/doc/bash
ubuntu@ubuntu: /usr/share/doc/bash$ cd ..
ubuntu@ubuntu: /usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu: /usr/share/doc$
```

### Step 13:

Use a relative path to change up one level from the current directory and then down into the dict directory:

```
cd ../dict
```

```
pwd
```

```
ubuntu@ubuntu: /usr/share/dict
ubuntu@ubuntu:/usr/bin$ cd /usr
ubuntu@ubuntu:/usr$ pwd
/usr
ubuntu@ubuntu:/usr$ cd /usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ cd bash
ubuntu@ubuntu:/usr/share/doc/bash$ pwd
/usr/share/doc/bash
ubuntu@ubuntu:/usr/share/doc/bash$ cd ..
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ cd ../dict
ubuntu@ubuntu:/usr/share/dict$ pwd
/usr/share/dict
ubuntu@ubuntu:/usr/share/dict$
```

#### Step 14:

To list the contents of the current directory, use the `ls` command:

```
cd
```

```
ls
```

```

ubuntu@ubuntu:/usr/bin$ cd /usr
ubuntu@ubuntu:/usr$ pwd
/usr
ubuntu@ubuntu:/usr$ cd /usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ cd bash
ubuntu@ubuntu:/usr/share/doc/bash$ pwd
/usr/share/doc/bash
ubuntu@ubuntu:/usr/share/doc/bash$ cd ..
ubuntu@ubuntu:/usr/share/doc$ pwd
/usr/share/doc
ubuntu@ubuntu:/usr/share/doc$ cd ../dict
ubuntu@ubuntu:/usr/share/dict$ pwd
/usr/share/dict
ubuntu@ubuntu:/usr/share/dict$ cd
ubuntu@ubuntu:~$ ls
Desktop    Downloads  Pictures   Templates  newname
Documents  Music      Public     Videos     snap
ubuntu@ubuntu:~$

```

I

In the output of the previous `ls` command, the file names were placed in a light blue color. This is a feature that many distributions of Linux automatically provide through a feature called an alias.

The color indicates what type the item is. The following table describes some of the more common colors:

Color	Type of File
<b>Black or White</b>	Regular file
<b>Blue</b>	Directory file
<b>Cyan</b>	Symbolic link file (a file that points to another file)
<b>Green</b>	Executable file (a program)

**Step 15:**

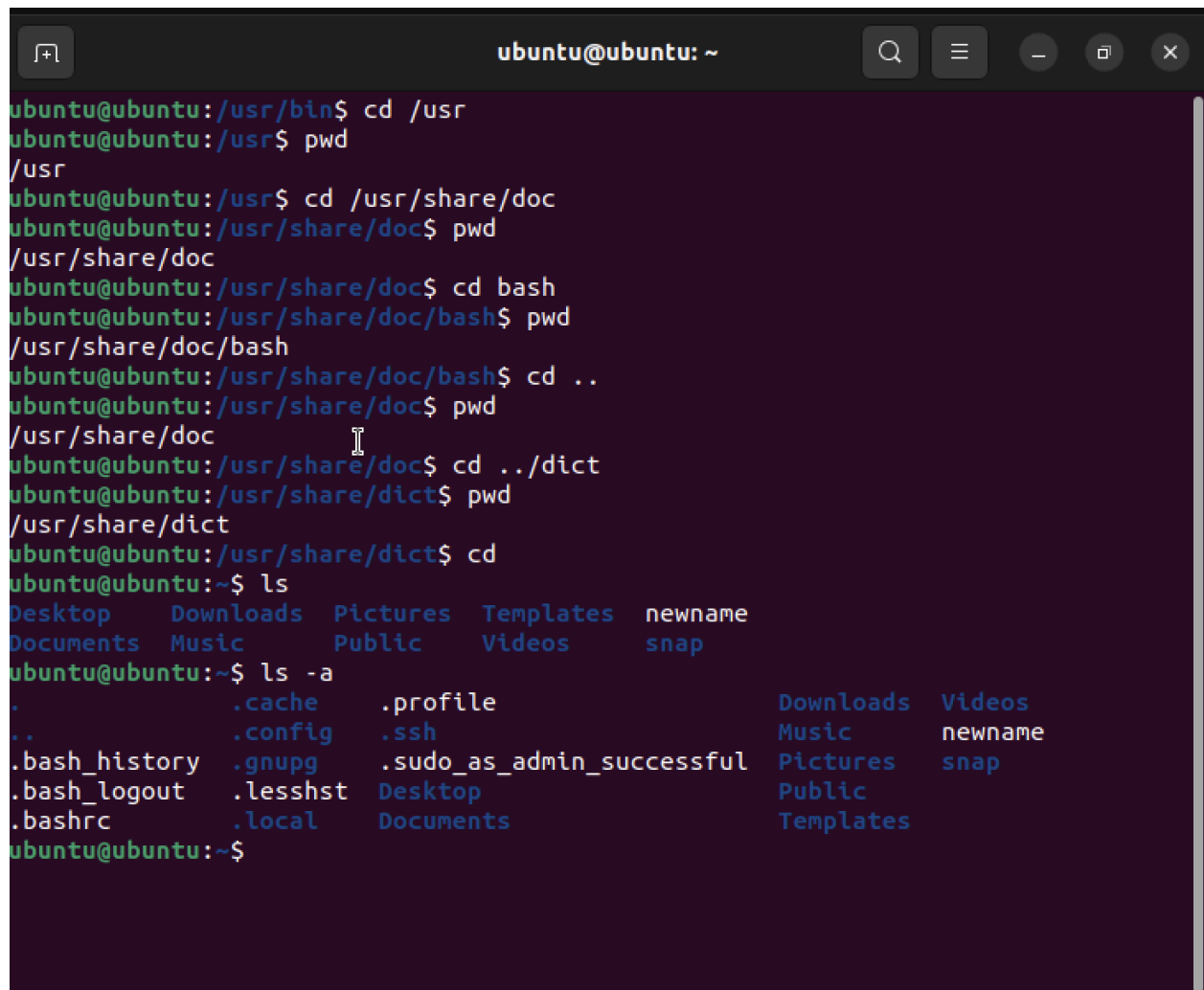


Not all files are displayed by default. There are files, called hidden files, that are not displayed by default. To display all files, including hidden files, use the `-a` option to the `ls` command:

```
ls -a
```

The names of hidden files begin with a period (a dot character). Typically, these files and often directories are hidden because they are not files you normally want to see.

**Two important "dot files" exist in every directory:** `.` (which represents the current directory) and `..` (which represents the directory above the current directory).

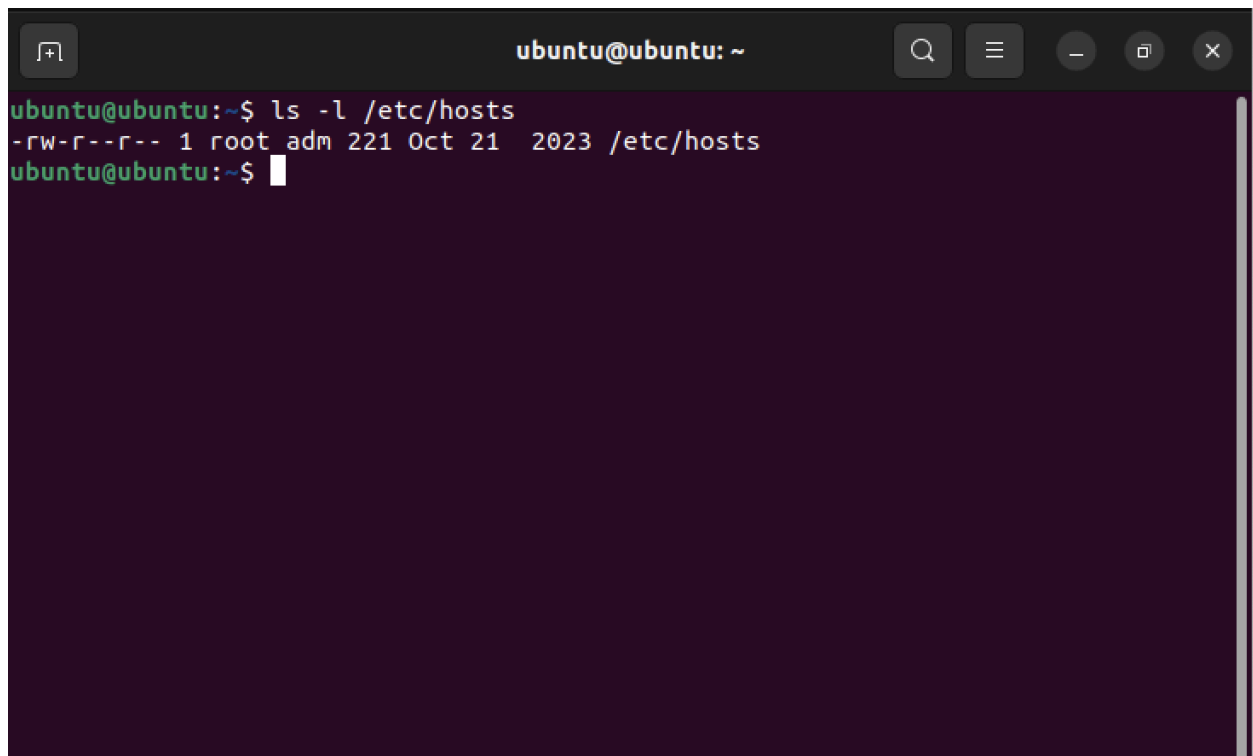


```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~/usr/bin$ cd /usr  
ubuntu@ubuntu:~/usr$ pwd  
/usr  
ubuntu@ubuntu:~/usr$ cd /usr/share/doc  
ubuntu@ubuntu:~/usr/share/doc$ pwd  
/usr/share/doc  
ubuntu@ubuntu:~/usr/share/doc$ cd bash  
ubuntu@ubuntu:~/usr/share/doc/bash$ pwd  
/usr/share/doc/bash  
ubuntu@ubuntu:~/usr/share/doc/bash$ cd ..  
ubuntu@ubuntu:~/usr/share/doc$ pwd  
/usr/share/doc  
ubuntu@ubuntu:~/usr/share/doc$ cd ../dict  
ubuntu@ubuntu:~/usr/share/dict$ pwd  
/usr/share/dict  
ubuntu@ubuntu:~/usr/share/dict$ cd  
ubuntu@ubuntu:~$ ls  
Desktop    Downloads  Pictures   Templates  newname  
Documents  Music      Public     Videos     snap  
ubuntu@ubuntu:~$ ls -a  
.  
..  
.bash_history  
.bash_logout  
.bashrc  
ubuntu@ubuntu:~$
```

## Step 15:

By itself, the `ls` command just provided the names of the files and directories within the specified (or current) directory. Execute the following command to see how the `-l` option provides more information about a file:

```
ls -l /etc/hosts
```

A terminal window titled 'ubuntu@ubuntu: ~' with search, menu, and window control icons in the title bar. The terminal shows the command 'ls -l /etc/hosts' being executed, resulting in the output: '-rw-r--r-- 1 root adm 221 Oct 21 2023 /etc/hosts'. The prompt 'ubuntu@ubuntu:~\$' is visible on the line below the output.

```
ubuntu@ubuntu:~$ ls -l /etc/hosts
-rw-r--r-- 1 root adm 221 Oct 21 2023 /etc/hosts
ubuntu@ubuntu:~$
```

Your output should be similar to the following:

```
sysadmin@localhost:~$ ls -l /etc/hosts
-rw-r--r-- 1 root root 150 Jul 28 15:18 /etc/hosts
sysadmin@localhost:~$
```

So, what does all of this extra output mean? The following table provides a brief breakdown of what each part of the output of `ls -l` means:

---

-            The first character, a - in the previous example, indicates what type of "file" this is. A -character is for a plain file while a d character would be for a directory.

---

**rw-r--r-**    This represents the permissions of the file.  
-

---

**1**            This represents something called a hard link count (discussed later).

---

**root**        The user owner of the file.

---

**root**        The group owner of the file.

---

**150**        The size of the file in bytes

---

**Jul 28**    The date/time when the file was last modified.  
**15:18**

### Step 16:

Sometimes you want to see not only the contents of a directory, but also the contents of the subdirectories. You can use the **-R** option to accomplish this:

```
ls -R /etc/udev
```

The **-R** option stands for "**recursive**". All of the files in the **/etc/udev** directory will be displayed as well as all of the files in each subdirectory, in this case the rules.d subdirectory.

Be careful of the **-R** option. Some directories are very, very large!

```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ ls -l /etc/hosts  
-rw-r--r-- 1 root adm 221 Oct 21 2023 /etc/hosts  
ubuntu@ubuntu:~$ ls -R /etc/udev  
/etc/udev:  
hwdb.d rules.d udev.conf  
  
/etc/udev/hwdb.d:  
  
/etc/udev/rules.d:  
70-snap.firefox.rules 70-snap.snapd.rules ubuntu--vg-ubuntu--lv.rules  
ubuntu@ubuntu:~$
```

### Step 17:

You can use file *globbing* (wildcards) to limit which files or directories you see. For example, the `*` character can match "zero or more of any characters" in a filename. Execute the following command to display only the files that begin with the letter `s` in the `/etc` directory:

```
ls -d /etc/s*
```

Note that the `-d` option prevents files from subdirectories from being displayed. It should always be used with the `ls` command when you are using file globbing.

```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ ls -l /etc/hosts  
-rw-r--r-- 1 root adm 221 Oct 21 2023 /etc/hosts  
ubuntu@ubuntu:~$ ls -R /etc/udev  
/etc/udev:  
hwdb.d rules.d udev.conf  
  
/etc/udev/hwdb.d:  
  
/etc/udev/rules.d:  
70-snap.firefox.rules 70-snap.snapd.rules ubuntu--vg-ubuntu--lv.rules  
ubuntu@ubuntu:~$ ls -d /etc/S*  
ls: cannot access '/etc/S*': No such file or directory  
ubuntu@ubuntu:~$ ls -d /etc/s*  
/etc/sane.d /etc/shells /etc/subuid  
/etc/security /etc/skel /etc/subuid-  
/etc/selinux /etc/snmp /etc/sudo.conf  
/etc/sensors.d /etc/speech-dispatcher /etc/sudo_logsrvd.conf  
/etc/sensors3.conf /etc/ssh /etc/sudoers  
/etc/services /etc/ssl /etc/sudoers.d  
/etc/sgml /etc/sss /etc/sysctl.conf  
/etc/shadow /etc/subgid /etc/sysctl.d  
/etc/shadow- /etc/subgid- /etc/systemd  
ubuntu@ubuntu:~$
```

### Step 18:

The `?` character can be used to match exactly 1 character in a file name. Execute the following command to display all of the files in the `/etc` directory that are exactly four characters long:

```
ls -d /etc/????
```

```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ ls -l /etc/hosts  
-rw-r--r-- 1 root adm 221 Oct 21 2023 /etc/hosts  
ubuntu@ubuntu:~$ ls -R /etc/udev  
/etc/udev:  
hwdb.d rules.d udev.conf  
  
/etc/udev/hwdb.d:  
  
/etc/udev/rules.d:  
70-snap.firefox.rules 70-snap.snapd.rules ubuntu--vg-ubuntu--lv.rules  
ubuntu@ubuntu:~$ ls -d /etc/S*  
ls: cannot access '/etc/S*': No such file or directory  
ubuntu@ubuntu:~$ ls -d /etc/s*  
/etc/sane.d /etc/shells /etc/subuid  
/etc/security /etc/skel /etc/subuid-  
/etc/selinux /etc/snmp /etc/sudo.conf  
/etc/sensors.d /etc/speech-dispatcher /etc/sudo_logsrvd.conf  
/etc/sensors3.conf /etc/ssh /etc/sudoers  
/etc/services /etc/ssl /etc/sudoers.d  
/etc/sgml /etc/sssd /etc/sysctl.conf  
/etc/shadow /etc/subgid /etc/sysctl.d  
/etc/shadow- /etc/subgid- /etc/systemd  
ubuntu@ubuntu:~$ ls -d /etc/????  
/etc/alsa /etc/dpkg /etc/ldap /etc/perl /etc/skel /etc/udev  
/etc/cups /etc/gdm3 /etc/mtab /etc/qemu /etc/snmp  
/etc/dhcp /etc/init /etc/newt /etc/sgml /etc/sssd  
ubuntu@ubuntu:~$
```

### Step 19:

By using square brackets [ ] you can specify a single character to match from a set of characters. Execute the following command to display all of the files in the **/etc** directory that begin with the letters a, b, c or d:

```
ls -d /etc/[abcd]*
```

```
ubuntu@ubuntu: ~  
ubuntu@ubuntu:~$ ls -d /etc/????  
/etc/alsa /etc/dpkg /etc/ldap /etc/perl /etc/skel /etc/udev  
/etc/cups /etc/gdm3 /etc/mtab /etc/qemu /etc/snmp  
/etc/dhcp /etc/init /etc/newt /etc/sgml /etc/sss  
ubuntu@ubuntu:~$ ls -d /etc/[abcd]*  
/etc/adduser.conf /etc/chatscripts  
/etc/adjtime /etc/cloud  
/etc/alsa /etc/console-setup  
/etc/alternatives /etc/cracklib  
/etc/anacrontab /etc/cron.d  
/etc/apg.conf /etc/cron.daily  
/etc/apm /etc/cron.hourly  
/etc/apparmor /etc/cron.monthly  
/etc/apparmor.d /etc/cron.weekly  
/etc/apport /etc/crontab  
/etc/appstream.conf /etc/cryptsetup-initramfs  
/etc/apt /etc/crypttab  
/etc/avahi /etc/cups  
/etc/bash.bashrc /etc/cupshelpers  
/etc/bash_completion.d /etc/dbus-1  
/etc/bindresvport.blacklist /etc/dconf  
/etc/binfmt.d /etc/debconf.conf  
/etc/bluetooth /etc/debian_version  
/etc/brlapi.key /etc/default  
/etc/brltty /etc/deluser.conf  
/etc/brltty.conf /etc/depmod.d  
/etc/ca-certificates /etc/dhcp  
/etc/ca-certificates.conf /etc/dictionaries-common  
/etc/ca-certificates.conf.dpkg-old /etc/dpkg  
ubuntu@ubuntu:~$
```

### Remember:

You are required to complete the lab and record your answers. Rename the file using your first name and the lab number, e.g. **washington3.1.docx**, and submit it through Canvas to receive marks for this lab.