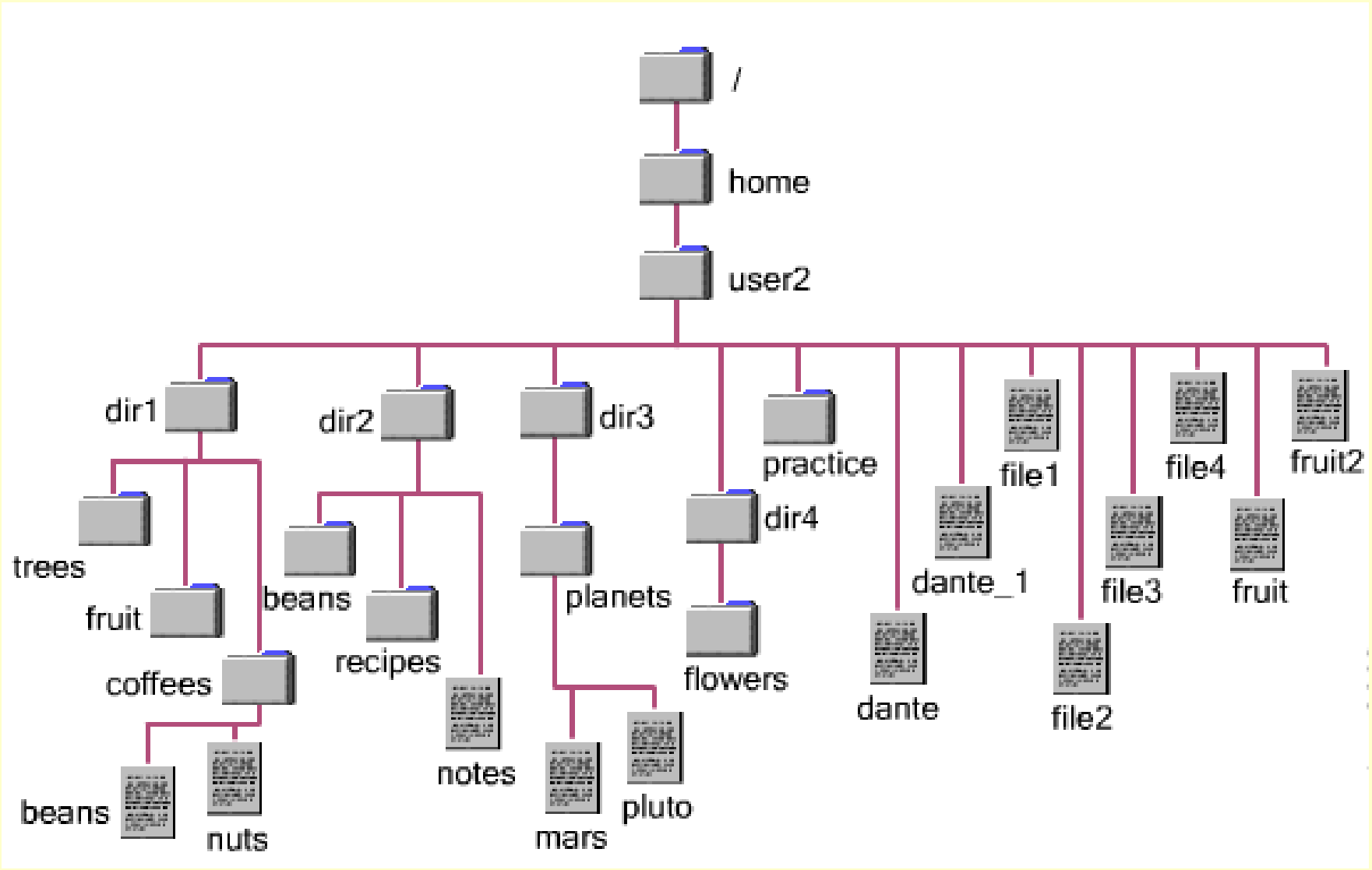
Server Operating Systems

Lecture 2
Accessing Files and Directories

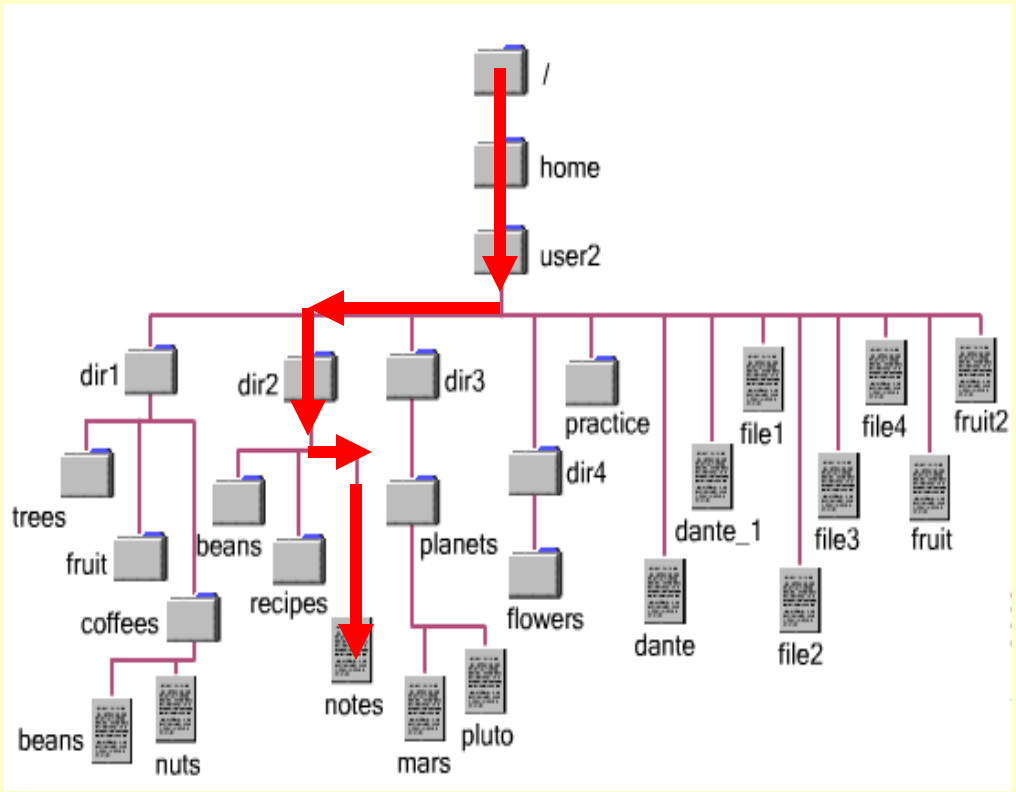
Accessing Files and Directories

- The File System
- Directory Paths
- Navigating the File System
- Listing Directory Contents
- Identifying and Using Metacharacters

Class File Tree Structure



Directory Structure



A *pathname* specifies the location of a file on disk.

Every file and directory on a system has a pathname.

The '/' separates file and directory names in a pathname.

What is the pathname of the file pointed to by the red arrows?

Types of Pathnames

Absolute Pathname

- Specifies the location of a file in relation to the *entire* file system.
- Always starts with the root (‘/’) directory.

Relative Pathname

- Specifies the location of a file in relation to the current directory.
- Can get to a file or directory in the current directory by simply using its name.
- Does not start with ‘/’.

Shorthand Directory Names

`~`	`(tilde)`	the user's home directory.
`.`	`(dot)`	The current directory.
`..`	`(dot dot)`	The parent directory.
`-`	`(dash)`	The last directory visited (Not on all shells).
`~username`		Another user's home directory. (If you are logged on as root)

What is the absolute pathname to the dir2 directory?

`/home/user2/dir2`

What is the absolute pathname to the planets directory?

`/home/user2/dir3/planets`

What is the absolute pathname of the notes file?

`/home/user2/dir2/notes`

From the student's home directory, what is the relative pathname to the dir4 directory?

`dir4`

If the student is in the dir2 directory, what is the relative pathname to the beans file?

`../dir1/coffees/beans`

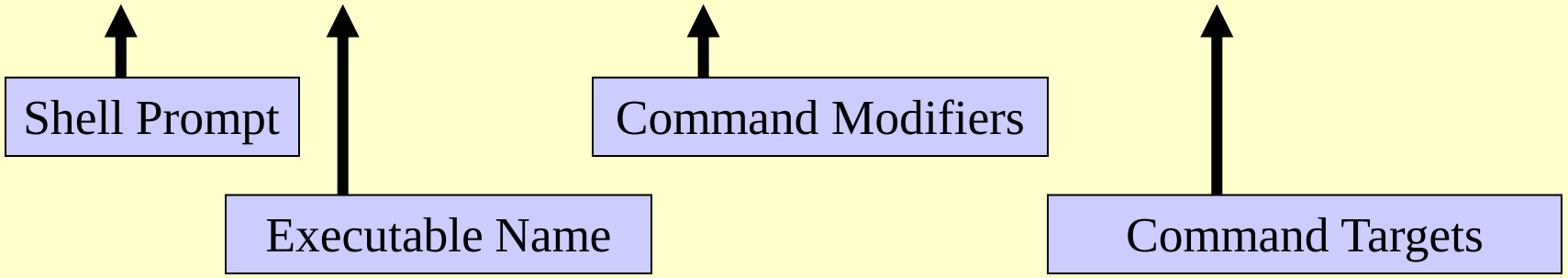
If the student is in the dir1 directory, what is the relative pathname to the practice directory?

`../../practice`

Unix Command-Line Syntax

Before you can use Unix commands effectively, you need to understand their general syntax.

```
$ command  [option(s)]  [argument(s)]
```



Example commands

```
$ cal

$ cal 2002                  one argument

$ cal 10 2008               two arguments

$ cal 9 1752                two arguments

$ man cal                   one argument

$ date

$ date -u                   one option

$ banner "hi there"         one argument
```

Unix Command-Line Syntax Rules

- Space is used as a delimiter between command parts
- Maximum of 256 characters on a single line
- Commands are lower case
- Options can be upper or lower case
- Options usually preceded by a hyphen
- Multiple options can be used with one hyphen
- Many commands do not require all three parts
- Multiple commands can be entered on one line if separated by a semicolon (;)

Displaying the Current Directory

`pwd` ("Print Working Directory")

will display the absolute pathname to the current directory.

Changing Directories

`cd` ("Change Directory")

to move to a different directory in the file system.

- Accepts absolute or relative pathnames.

What do these mean?

```
cd ~
cd .
cd ..
cd -
```

The `ls` Command

`ls`

Gives a listing of the contents (files and directories) of a directory.

Syntax: **`ls [-option(s)] [pathname(s)]`**

- With no options you get a “wide” listing.
- Sorted alphabetically by default.

This is the way to find out what is in your file system.

Hidden Files

Any file whose name begins with a '.' is a *hidden file* in Unix.

- Not shown with **`ls`** by default.
- The links to '.' and '..' are hidden files.

To see hidden files in Unix,

- Use **`ls -a`** to show "all" files.
- Use **`ls -A`** to show all files except '.' and '..'

Look in your home directory with the GUI file manager. Can you see the hidden files? Look at the view menu - can you make them visible?

Displaying File Types

Use **`ls -F`** to get a “wide” listing with indicators as to the type of each file.

File Type	Symbol	Notes
Directory	/	Forward Slash
ASCII Text File	(none)	No symbol
Executable	*	Asterisk
Symbolic link	@	The at Sign
Named pipe	|	Pipe Symbol (vertical bar)
Socket	=	Equal Sign

Displaying a Long Listing

`ls -l`

- This gives a “long” listing of directory contents.
- Sorted alphabetically by default. Can use **`ls -lt`** to sort by time.

File Types with `ls -l`

The “long” listing shows the file type at the far left of the listing.

- Most common are ‘-’ for a regular file and ‘d’ for a directory.

Two codes, ‘b’ and ‘c’ are shown with **`ls -l`** but *not* **`ls -F`**.

- ‘b’ stands for *Block Device*, a hardware device file that transfers data in blocks larger than one byte.

- ‘c’ stands for *Character Device*, a hardware device file that transfers data one byte at a time.

Listing Individual Subdirectories

Use **`ls –ld`** to find out information about a directory itself, *not* the contents of it.

You can use the “recursive” option **`ls –R`** to list a directory and all of its subdirectories.

- Can produce a lot of output.
- the **`-r`** and **`–R`** options are used with many Unix commands.

Identifying Metacharacters

A *metacharacter* is any keyboard character that has a "special" meaning to a shell.

The three most commonly used metacharacters in Unix are: *, ?, ; and [ ].

Using Metacharacters

* (Asterisk)
  - '*' matches zero or more characters.
    - Except the leading dot '.' on a hidden file.
  - Referred to as a *wildcard* character.
  - Using with the **`ls`** command will list all the files that match the pattern made by using '*', as well as the directories and their contents that match.

Using Metacharacters

? (Question Mark)

- Matches a single character.
  - Except the leading dot '.' on a hidden file.

- To match "one character or another," use this syntax: **`(c1|c2)`** where `c1` and `c2` are two different single characters.

Using Metacharacters

[ ] (Square Brackets)

- Match one of a range or one of a set of specified characters that could be in *one* character's position.
- Characters inside the brackets do not need to be in any order.
- If looking for a range, then the characters must be in order.
  - The '-' is used to specify a range. [a-z] looks for any lower case character between 'a' and 'z' inclusive.
- Case sensitive. [a-z] and [A-Z] are *not* the same!

Using Metacharacters

; (Semicolon)

- Enables typing in multiple commands on the command line without having to press “enter” in between them.
- Put the ‘;’ in between each command.
- Called the “*command separator*.”

Metacharacters Exercise

What does this command do?

```
$ ls  *[1-5]*p
```