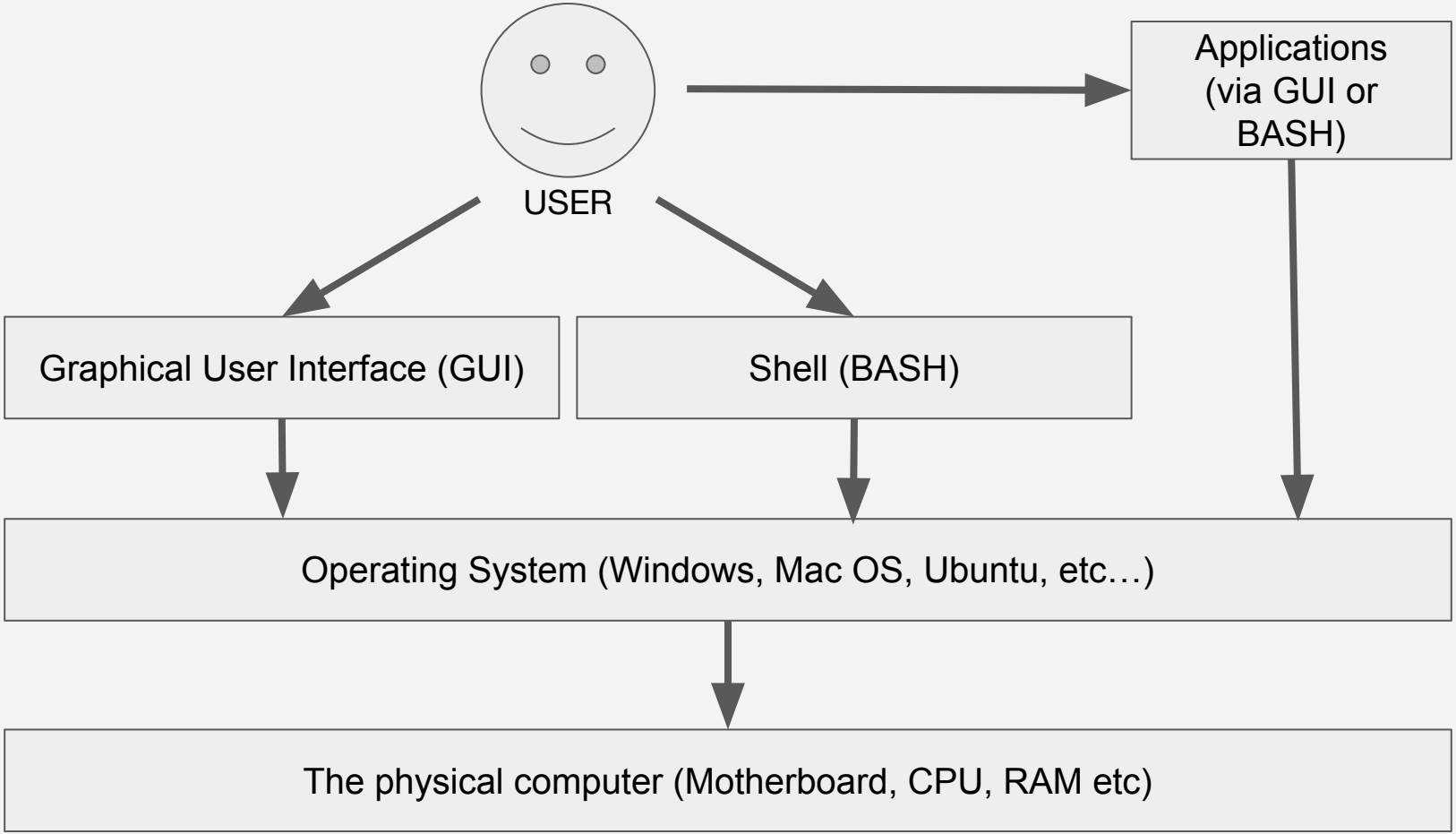
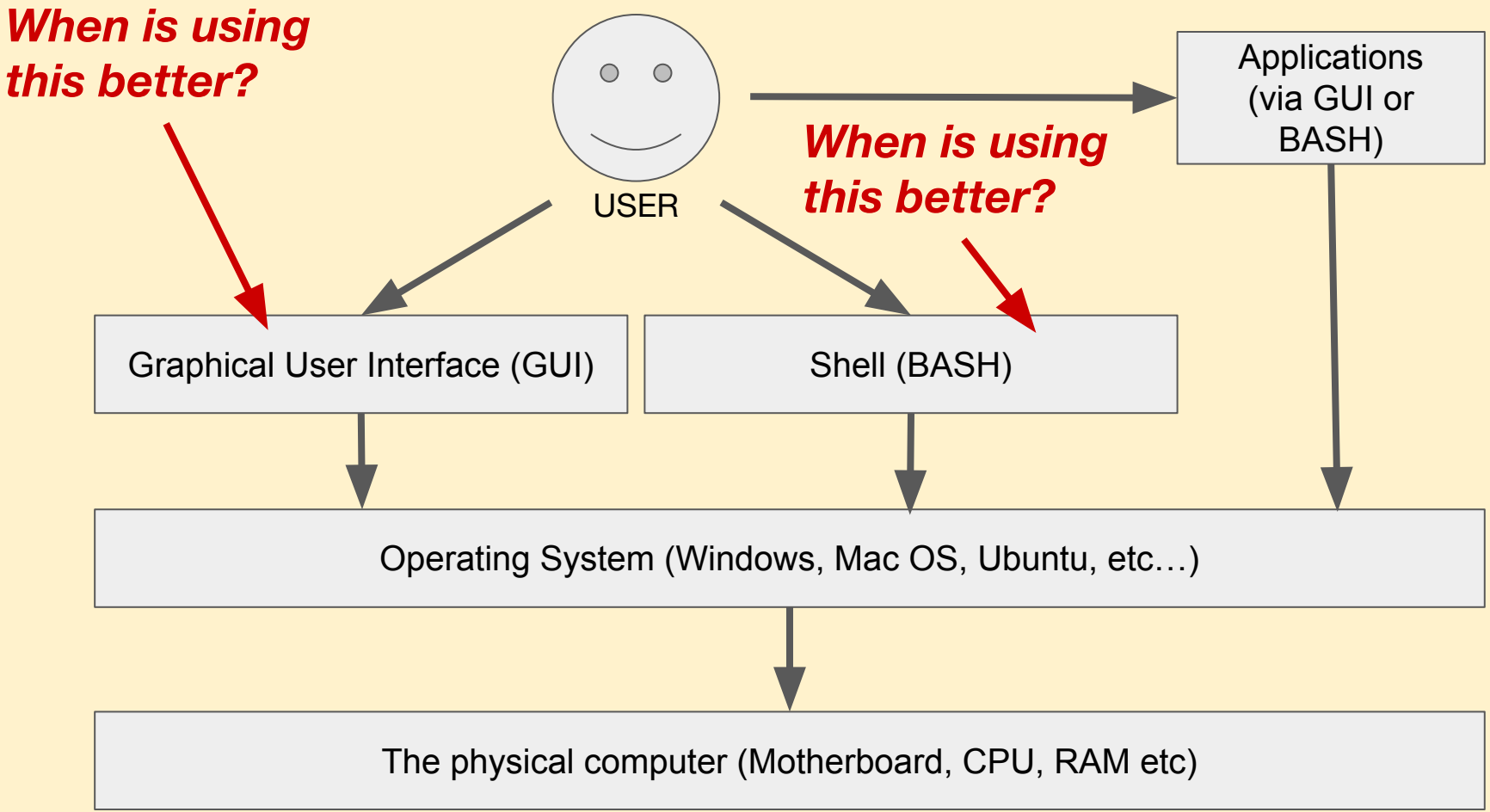
CSc 352
UNIX, files, and bash

Benjamin Dicken

Announcements

- Discord Server
- PA 1
- Did you purchase the textbook?


USER
Applications (via GUI or BASH)
Graphical User Interface (GUI)
Shell (BASH)
Operating System (Windows, Mac OS, Ubuntu, etc…)
The physical computer (Motherboard, CPU, RAM etc)


When is using this better?
USER
Applications (via GUI or BASH)
When is using this better?
Graphical User Interface (GUI)
Shell (BASH)
Operating System (Windows, Mac OS, Ubuntu, etc…)
The physical computer (Motherboard, CPU, RAM etc)

UNIX - What is it?

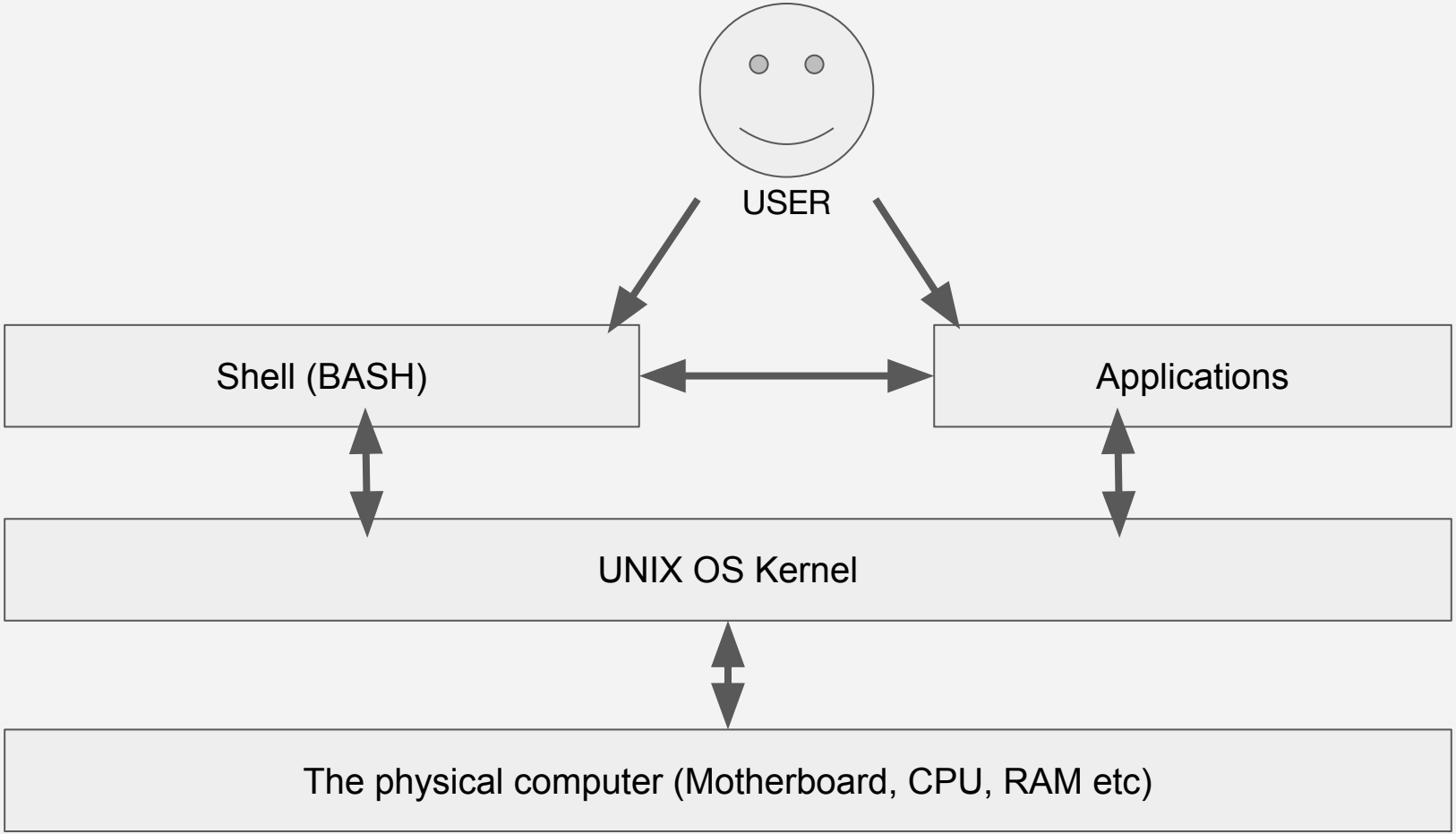
- **Unix** is an operating system (OS)
- **Linux** is a family of Unix-like OSs, and based on the Linux kernel
- A **Unix-like** OS is one that has a similar design to the original unix, but might have little (or none) of the original Unix codebase.

    - Mac OS is a **Unix-like** OS
    - Ubuntu, Debian, Fedora are **Linux** OSs
    - Windows on its own is none of the above, though you can install the WSL

UNIX - What is it?

- **Unix** is an operating system (OS)
- **Linux** is a family of Unix-like OSs, and based on the Linux kernel
- A **Unix-like** OS is one that has a similar design to the original unix, but might have little (or none) of the original Unix codebase.

    - Mac OS is a **Unix-like** OS
    - Ubuntu, Debian, Fedora are **Linux** OSs
    - Windows on its own is none of the above, though you can install the WSL


USER
Shell (BASH)
Applications
UNIX OS Kernel
The physical computer (Motherboard, CPU, RAM etc)

The Shell

- A Unix **shell** is a text-based command processing program
  - Gives users ability to run commands, control computer, run apps
- Many options (sh, zsh, ksh, bash)
- For this course, we will use bash, which is the default for Lectura

```
echo $SHELL
```

Bash commands

- Bash provides an “infinite loop” of waiting for and processing commands with the syntax

command_name arguments

- ***command_name*** - the name of the command to be run (ls, pwd, cd …)
- ***arguments*** - options / input to determine how the command should work (like function arguments)
- Type commands, then press **ENTER** to begin

Running a command

- Connect to lectura, get to the shell (Bash)
- Run the following commands:

```
whoami
ls
cal
```

- What do these do?

Running a command

- Now run some commands with arguments

```
whoami
ls -l
cal 10 2020
```

- What do these do?

Command Line options

Most commands have options, come in a few types

Flag

A boolean option that begins with a dash (enable/disable feature)

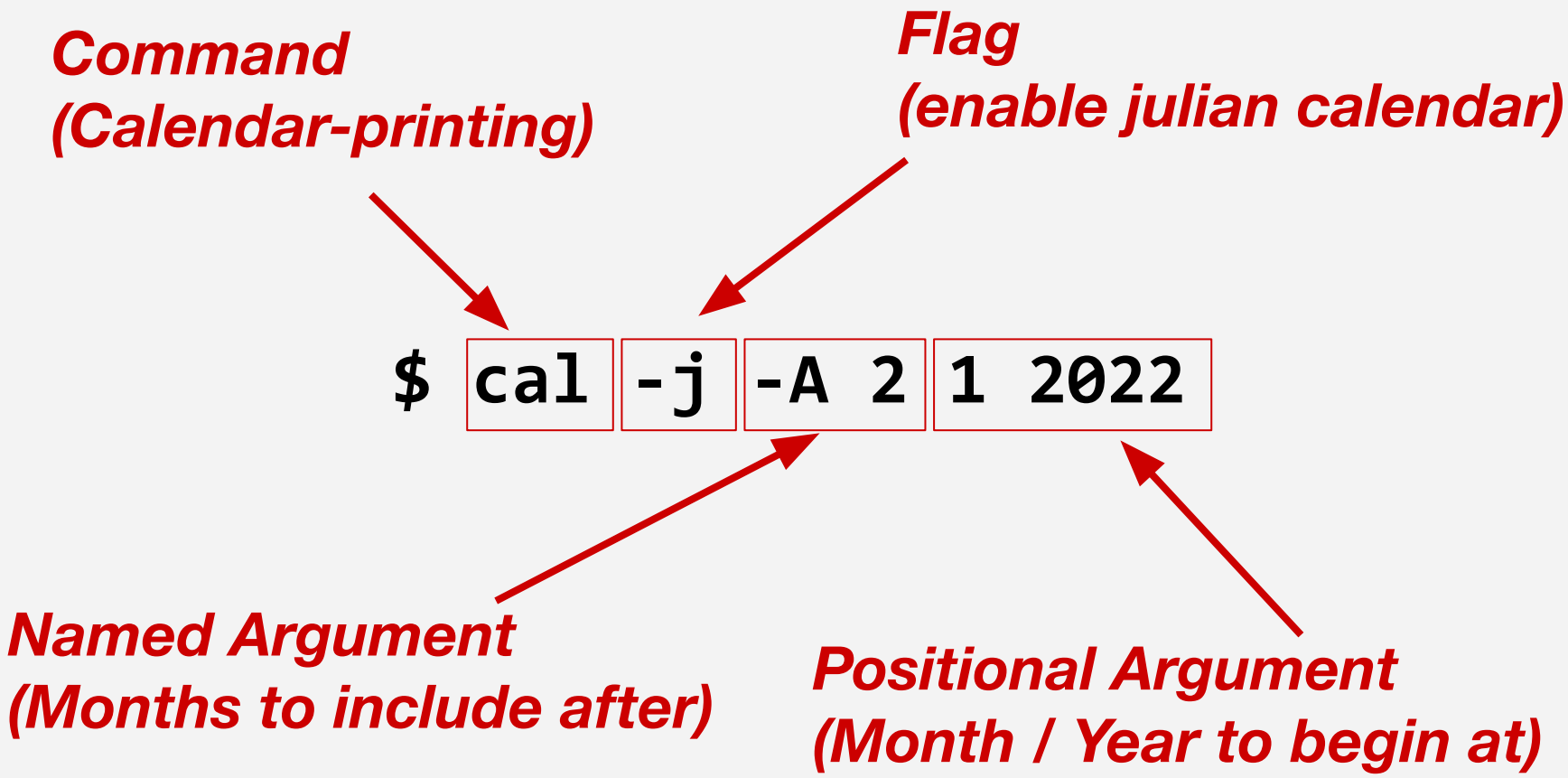
Named argument

A flag, with a value following

Positional argument

An argument with no flag

```
$ cal -j -A 2 1 2022
```

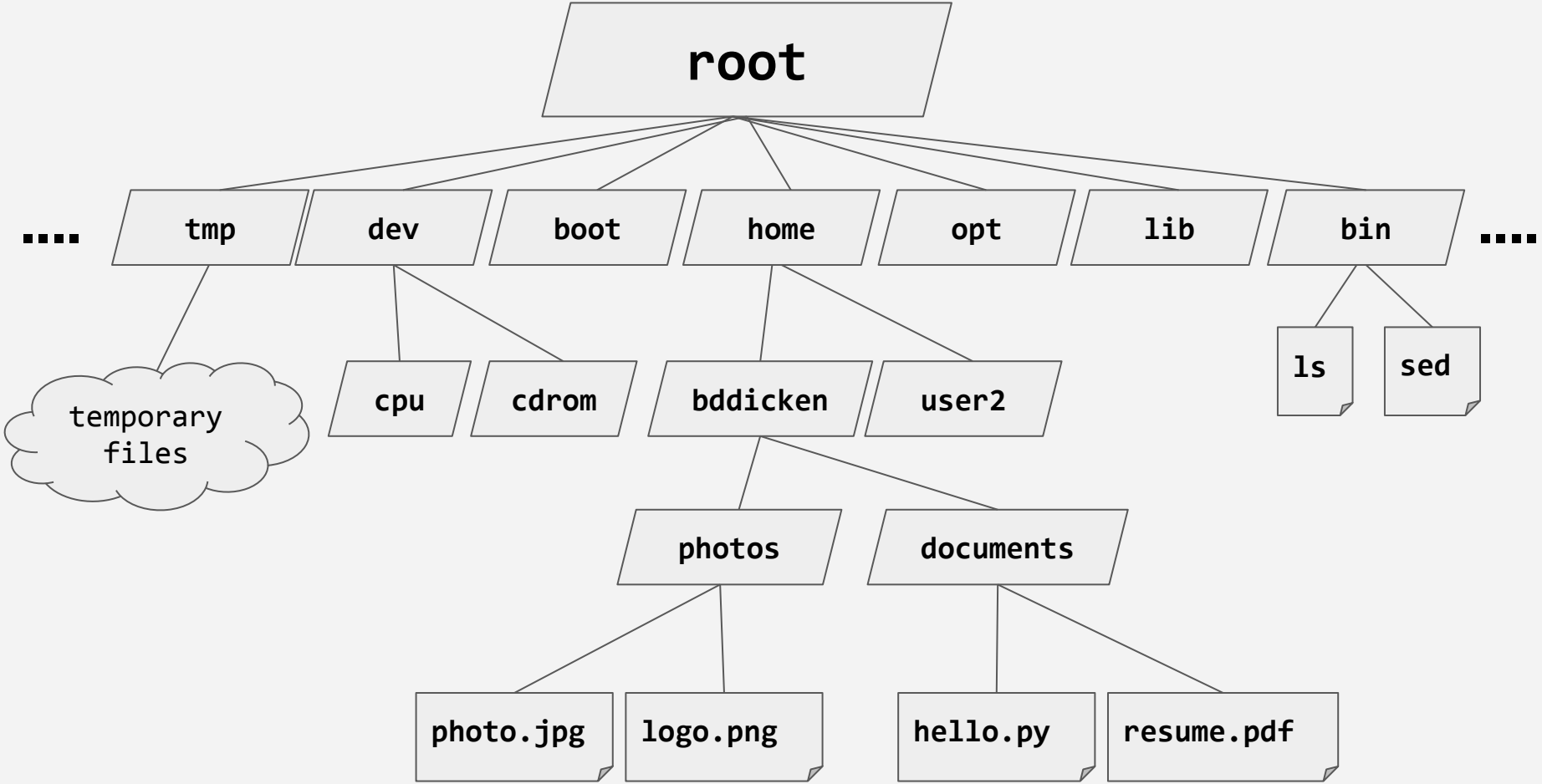

Command
(Calendar-printing)
Flag
(enable julian calendar)
$ cal -j -A 2 1 2022
Named Argument
(Months to include after)
Positional Argument
(Month / Year to begin at)

The File System

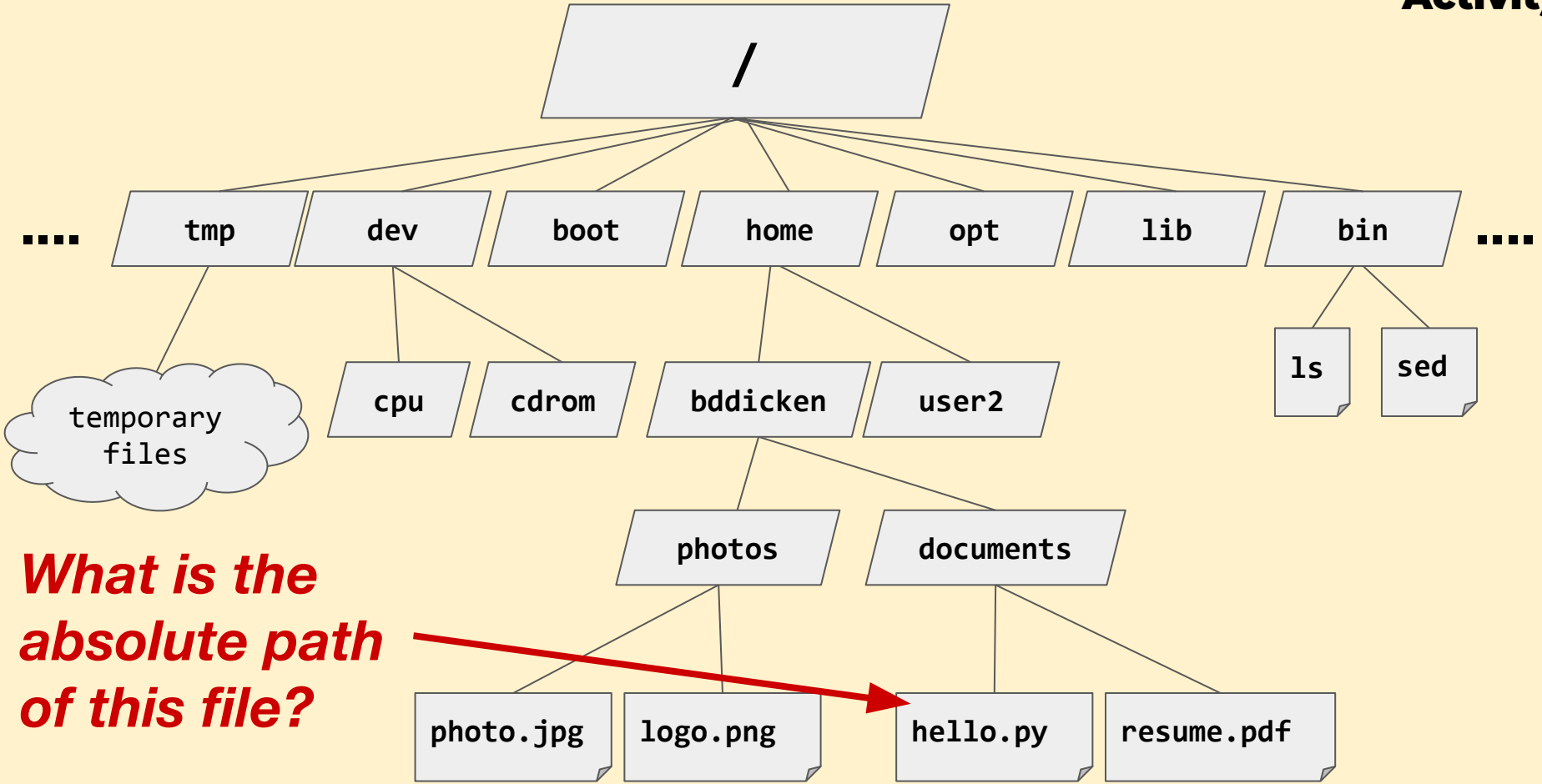
- A **File System** is a software system that defines how files are organized, stored, and retrieved on a hard drive
- Many various implementations
- Most Unix(-like) systems share a somewhat standardized, tree-like file structure

Files

- “Everything is a File”
- Directories (folders) are just a special type of file, other files within
- Hardware files
- Files are sequences of bytes (1s and 0s) with some additional context

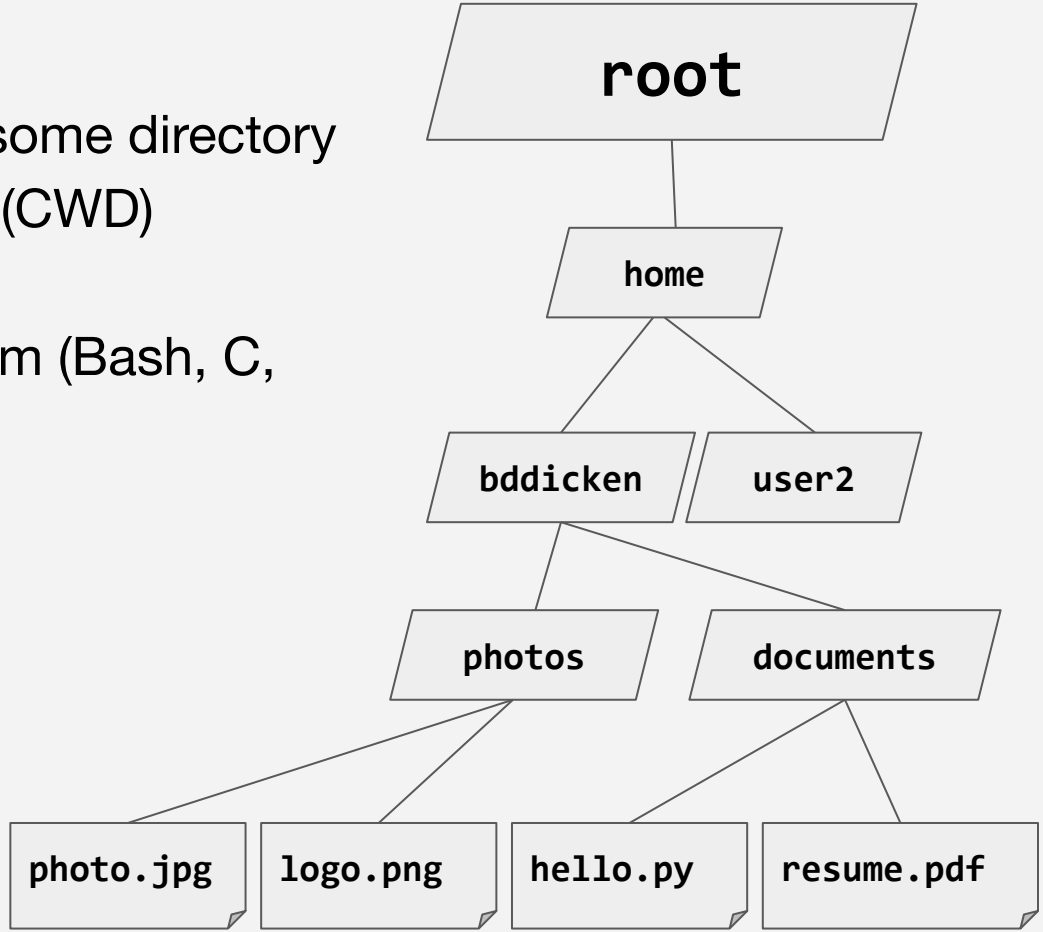

root
....
tmp
dev
boot
home
opt
lib
bin
....
temporary files
cpu
cdrom
bddicken
user2
ls
sed
photos
documents
photo.jpg
logo.png
hello.py
resume.pdf

Activity

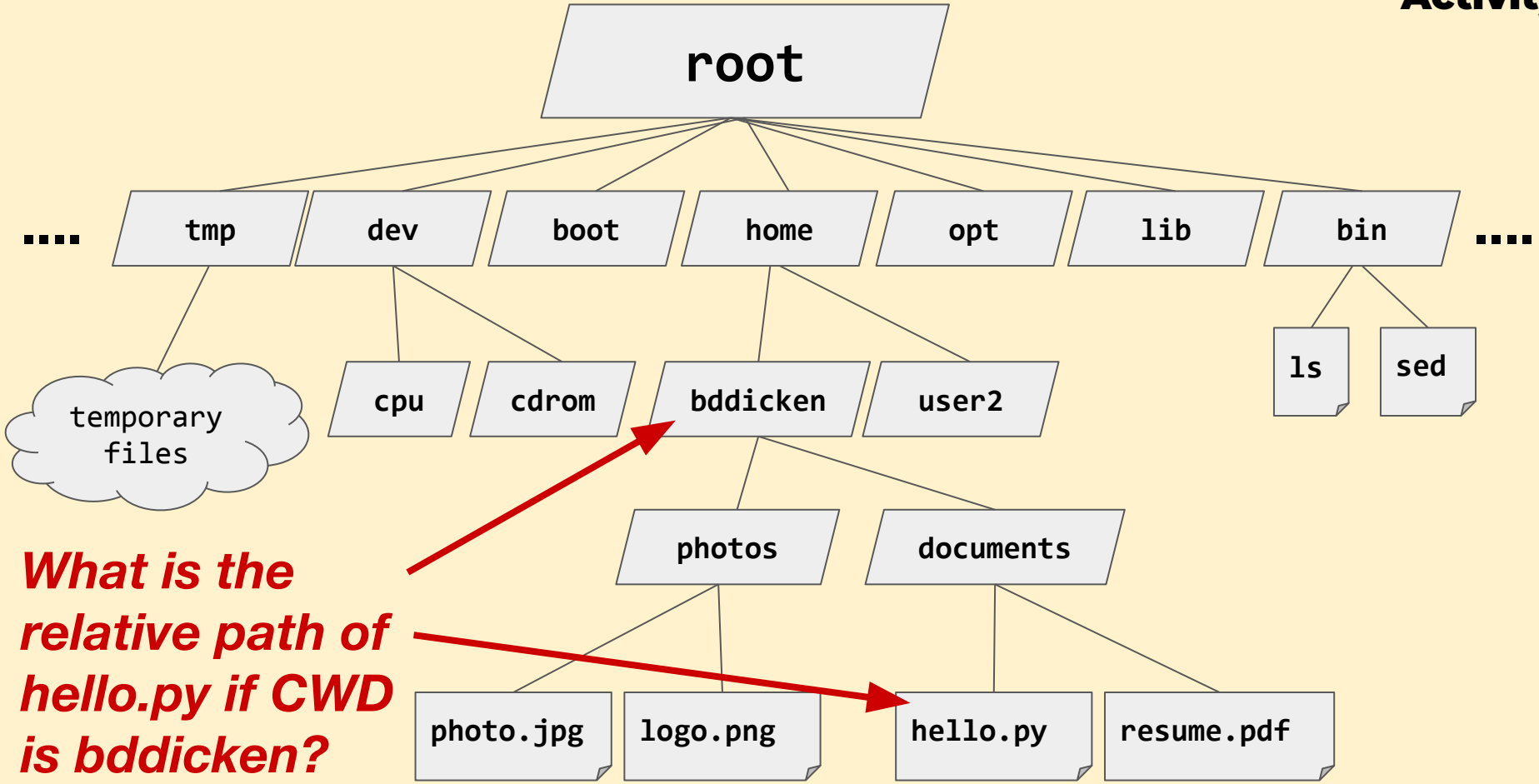


What is the absolute path of this file?

- In **Bash** you are always “in” some directory
- **Current-Working Directory** (CWD)
- Use the **`pwd`** command
- In Unix, every running program (Bash, C, python, java) has a **CWD** too
- Relative Paths
- Home directories



Activity



What is the relative path of hello.py if CWD is bddicken?

Command Line options

For navigating the files system

```
cd    ls    find
```

Commands: Files and the File System

`cd` - change directory

- Absolute or relative paths
- Special symbols: **`~  ..  .  /`**

`ls` - list directory content

- CWD or by path
- Many options: **`-a -l -R ....`**

Many more

`cp  mv  rm  touch  mkdir  find  cat ....`

TLCL Chapters 2, 3, 4

Write the commands to:

1. Log into lectura
2. Create a directory named **352-py-test**
3. Within that directory, create a file named **testing.py**
4. Put **print('test')** in it
5. Run the python code!

Use: **`ssh` `mkdir` `cd` `touch` `nano` `python3`**

(If you don’t have a computer, work with another, or do on-paper)

Commands: File Editing

Variety of in-shell text editors - can use to write code!

Vim (my preference)

```
$ vim code.c
```

Nano (good for beginners)

```
$ nano code.c
```

Emacs (Not my primary editor, but you are welcome to use / learn)

```
$ emacs code.c
```

Command: SCP

SCP = Secure Copy

Copy files between computers over a network
VERY useful for this course, copying to / from Lectura

```
$ scp from_location to_location

$ scp my_code.c bddicken@lectura.cs.arizona.edu:352/pa1/

$ scp -r  pa1 bddicken@lectura.cs.arizona.edu:352
```

Command: Learning about commands

The **man** command can bring up **manual** pages for various commands on a system

```
$ man man

$ man ls

$ man ssh
```

Input / Output

Program running on a Unix machine have three types of I/Os

Standard Input (stdin)

You've used this before! The **`input()`** function in python

Standard Output (stdout)

The text output your program produces (not including file I/O)

Standard Error (stderr)

A Special type of output for error messages only

Input / Output

Program running on a Unix machine have three types of I/Os

Standard Input (stdin)

Bash default: typed text

Standard Output (stdout)

Bash default: printed out

Standard Error (stderr)

Bash default: printed out

Controlling std in / out / err in bash

```
$ command > file
$ command < file
$ command >> file
$ command 1> file
$ command 2> file
$ command &> file
$ command1 | command2
```

TLCL Chapter 6

Controlling std in / out / err in bash

Write bash commands to do the following:

1. Save the names of all the files / directories on the root drive of the computer to a file named **root.txt**
2. Print the calendar of the current month, but only show the line with the day of the week

Files

Files have metadata

size, permissions, owner, creation date

```
$ man ls
$ ls -ltrsh
```

```
$ ls -ltrsh
total 148M
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Templates
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Music
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Videos
 14K drwxr-xr-x  3 bddicken bddicken   10 Sep  4  2012  pages
7.0K -rw-r--r--  1 bddicken bddicken  427 Sep 23  2012  id_rsa.pub
 14K drwxrwxrwx 12 bddicken bddicken   13 Oct 15  2012  android-sdk-linux
 14K drwxr-xr-x  9 bddicken bddicken    9 Feb 15  2013  eclipse_workspace
 14K drwxr-xr-x  3 bddicken bddicken    9 Jun 10  2013  Pictures
 512 drwxr-xr-x  4 bddicken bddicken    4 Jun 28  2013  workspace
 512 drwxrwxr-x  3 bddicken bddicken    3 Sep  4  2013  R
 512 drwxr-xr-x  2 bddicken bddicken    3 Sep 18  2013  Public
 ....
```

See: https://mason.gmu.edu/~montecin/UNIXpermiss.htm

```
$ ls -ltrsh
total 148M
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Templates
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Music
 512 drwxr-xr-x  2 bddicken bddicken    2 Aug 27  2010  Videos
 14K drwxr-xr-x  3 bddicken bddicken   10 Sep  4  2012  pages
7.0K -rw-r--r--  1 bddicken bddicken  427 Sep 23  2012  id_rsa.pub
 14K drwxrwxrwx 12 bddicken bddicken   13 Oct 15  2012  android-sdk-linux
 14K drwxr-xr-x  9 bddicken bddicken    9 Feb 15  2013  eclipse_workspace
 14K drwxr-xr-x  3 bddicken bddicken    9 Jun 10  2013  Pictures
 512 drwxr-xr-x  4 bddicken bddicken    4 Jun 28  2013  workspace
 512 drwxrwxr-x  3 bddicken bddicken    3 Sep  4  2013  R
 512 drwxr-xr-x  2 bddicken bddicken    3 Sep 18  2013  Public
 ....
```

Patterns and Globbing

`*` will match any character(s)

```
$ ls *.c
$ wc bddicken*.txt > out.txt
```

`[. . .]` will match characters are within the brackets

```
$ ls code.[a-z]
$ mv tasks-[1-9].txt /home/bddicken/taskdir/
```

Controlling std in / out / err in bash

Write bash commands to do the following:

1. Read the **man** pages for **sort, uniq, head, tail, grep, cut**
2. MEDIUM: Get the alphabetically last word from `/usr/share/dict/words` that contains `'ii'`
3. HARDER: Get the first letters of the words from `/usr/share/dict/words` that contain the sequence `'idi'` and the letter `'z'`

Controlling std in / out / err in bash

```
$ cat /usr/share/dict/words | grep ii | tail -1

$ cat /usr/share/dict/words | grep idi | grep z | cut -c1 | uniq
```

Future UNIX topics

If time permits, throughout the course we can delve deeper into these unix topics:

- Users, groups, permissions
- Processes and process control
- Unix files, sym links, etc