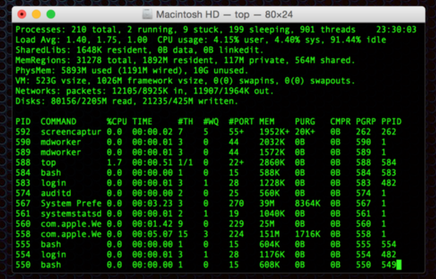
Basic UNIX

Not Really in Textbook

Terminals

https://en.wikipedia.org/wiki/File:DEC_VT100_terminal.jpg

- When computers were very large and expensive, people used **terminals** to connect to them
- These terminals were physical devices that resemble computers as we know them today
- Commands for the computer were typed into the terminal
- Nowadays, we often use software terminals in the place of physical terminals



https://en.wikipedia.org/wiki/File:Appleterminal2.png

Shells

- **Shells** are the software that run in terminals
- It's what we type commands into
- Various different shells, but **Bash shell** is one of the most common
- The terms terminal / command-line / console / shell are sometimes used interchangeably

Commands

- Commands are actually programs
- We run them by simply typing the name of the command:
  - Prompt$ ***command_name***
- Commands, and UNIX in general, are case sensitive!

Switches

- Many commands have options and parameters that can be indicated when we run the command
- These are often called **switches**
- Switches usually start with a **-** followed by a letter
  - ***command_name* -v**
- Sometimes the switch might be **--** followed by a word
  - ***command_name* --version**

- Some switches expect items after them
  - For example, a switch to specify a file
    - ***command_name* -f *file_name*.txt**
- Meaning of the same letter often differs between different commands

ls

- **ls** stands for list
- **ls** lists the files and directories in the current directory we're in
- **-l** provides more detailed information about each item in the list
- **-a** includes hidden items
  - In Unix, names that start with **.** are hidden by default

cd

- **cd** stands for change directory
- Used for moving around the file system
  - **cd *directory_name***

Root Directory

- On Windows, the top most folder is usually **C:**
- In Unix, **/** is the top most directory
- **/** is called the **root directory**
- Everything (all files and other directories) are contained inside of **/**, either directly or indirectly

Home Directory

- Each user has their own directory
- This directory is called the **home directory**
- A short cut for a user's **home directory** is **~**
- For example, "**cd ~**" will take a user to their **home directory**, regardless of where they are currently

Current Directory

- The **current directory** is the directory that we're presently inside
- A special symbol for referring to the **current directory** is **.**
  - Useful to specifically refer to something in the current directory
    - ***./program_name***

Parent Directory

- The **parent directory** is the directory that contains the current directory
- A special symbol for referring to the **parent directory** is **..**
- Very useful for going up / back a directory
  - **cd ..**

file

- File extensions are not as meaningful in Unix as in Windows
- Not uncommon for Unix files to have no extension
- **file** is a command that tells us the type of a file
  - **file *file_name***

less

- **less** is a program to view a files, particularly text and source code files
  - **less *file_name***
- Pressing arrow keys scrolls up and down in the file
- Pressing **q** exits **less**

cp

- **cp** stands for copy
- It's a way we can copy a file in UNIX
  - **cp *file_name new_file_name***

mv

- **mv** stands for move
- It's a way we can move a file
- It's also the way to rename a file
  - **mv *file_name new_file_name***

rm

- **rm** stands for remove
- It's a way we can delete a file
  - Note that rm does not send the file to a recycle or trash bin like in some OSes
- Use with caution!
  - **rm *file_name***

- If you try to use **rm** on a directory, get an error
- To remove a directory (and the contents inside it), need the **-r** switch:
  - **rm -r *directory_name***
  - **-r** stands for **recursive**
- **rm -r** can be a particularly dangerous command

pwd

- **pwd** stands for path of working directory
- Tells what your current directory is

mkdir

- mkdir stands for make directory
- Command for creating a new directory
  - **mkdir *directory_name***

man

- **man** stands for manual
- Really useful for finding the proper usage for a command
- Great way to find out the switches and option a command has
  - **man *command_name***

exit

- When you're done, you can logout by typing:
  - **exit**

nano

- **nano** is a simple text editor
  - Can be used for source code too
- To start, type the command and the file to open:
  - **nano *file_name***
  - If the file exists, it's opened
  - If the file does not exist, it'll be created

- To **save** (WriteOut) a file:
  - **^O** means press Ctrl and O at the same time
- To **exit**:
  - **^X** (press Ctrl and X at the same time)

vi

- **vi** is a popular text / source code editor
- Controls rather different
- vi has a concept of modes, specifically an insert and a command mode
  - **Insert mode** allows for typing text
  - **Command mode** allows for saving and exiting the program

- Type the command and the file name to open the file:
  - **vi *file_name***
  - If the file exists, its contents are opened and displayed
  - If it doesn't exist, it will be created
- Starts in command mode
- When in command mode, press **i** to go into insert mode
  - Text can be typed into the file then

- When done typing text, press **Esc** to go back to command mode
  - **:wq** saves the file and exits the program
  - **:q** quits
  - **:q!** quits without saving

Compiling Java Programs

- A Java file can be compiled by calling **javac** on the .java file:
  - **javac *ClassName*.java**

Running Java Programs

- A Java program can be ran by using the **java** command on a class name contained in .class file:
  - **java *ClassName***

Output Redirection

- Some programs output a lot of text to the console
- We can redirect this output into a file using **>** :
  - ***command* > *file_name*.txt**
  - If the file does not exist, it will be created
  - If already exists, overwrites it
    - **>>** can be used to instead append

Input Redirection

- Similar to output redirection, we can direct text from a file as input into a program using **<** :
  - ***command < file_name*.txt**

Piping

- **Piping** is redirecting the output of one command as the input to another command
- Way to "chain" or "link" commands together
- To pipe the output from ***command1*** into the input of ***command2***, separate them with a **|** :
  - ***command1* | *command2***

Useful Tips

- Pressing the **up arrow** puts your previous command
  - **Down arrow** goes back to more recent commands
- Pressing **tab** autocompletes commands
  - Useful for typing directory or file names