1 LECTURE SHELL

ar - augmented reality, vr - virtual reality
80% graphical user interface, ar/vr interfaces, text interfaces
Shell - text interface, that exposes operating systems's service,

$ - not the root user
~ - home or $HOME

command argument

```
missing:~$ echo hello
hello
```

Argument with spaces "hello world"

Navigation

A path on the shell
On linux and mac os /
On windows \

On linux and mac os / is the root of the file system
On windows there is one root for each partition c:\

A path that starts with / is absolute any other relative

Relative path is relative to the current working directory might be seen with (pwd command)
. - refers to the current working directory
..- refers to parent directory

ls - prints the contents of a given directory

Most commands accept flags that start with - to modify their behaviour

permissions

```
missing:~$ ls -l /home
drwxr-xr-x 1 missing  users  4096 Jun 15  2019 missing
```

--- --- --- - owner group other
d - directory
rwx - read write execute permission

r-x - read and execute only
- missing permission
mv - move
cp - copy
mkdir - make new directory
[OPTION]... [FILE]... - zero or more options zero or more files or directories
man - an interface for the system reference manuals
echo - display a line of text
sudo - do super user
man - manual

Connecting programs

Shell programs have to streams - input and output
Normally a program's input and output stream is the terminal
< > - rewire the direction of streams to a program
Example of redirection of streams

```
missing:~$ cat < hello.txt > hello2.txt
missing:~$ cat hello2.txt
hello
```

```
[me@linuxbox me]$ sort < file_list.txt > sorted_file_list.txt
```

>> append to a file
| - pipe, lets to chain programs
If the file starts with . then it's hidden
pwd - print working directory
cat - concatenate and display content

Homework 1

curl - transfer data from or to server
sh - command interpreter
shebang - char sequence of !# that process the rest of the line as interpreter directive, so the system uses the interpreter specified in the shegang and takes as argument the path of the current file
/bin - historically a container literally binary
rm - accepts also paths as arguments(absolute and relative)

2 LECTURE SHELL SCRIPTING

globbing - ?
- history instead of arrow up

'' - literal string

“” - format string

$

foo=bar
$foo - access value of a variable
$0-9 - argv

- `$0` - Name of the script
- `$1` to `$9` - Arguments to the script. `$1` is the first argument and so on.
- `$@` - All the arguments
- `$#` - Number of arguments
- `$?` - Return code of the previous command
- `$$` - Process identification number (PID) for the current script
- `!!` - Entire last command, including arguments. A common pattern is to execute a command only for it to fail due to missing permissions; you can quickly re-execute the command with sudo by doing `sudo !!`
- `$_` - Last argument from the last command. If you are in an interactive shell, you can also quickly get this value by typing `Esc` followed by `.` or `Alt+.`

Bash env directive

#!/usr/bin/env bash

Short circuit operators

&& - and operator
|| - or operator
; - separate commands

Evaluate expression

[[expression]]

Loops,conditions, std

Syntax of a while loop:
while condition;do
code…
Done

if condition; then
code…
else
fi

Stdin 0
Stdout 1
stderr 2
Example or redirecting stderr to null - 2>dev/null

Finding

```
# Find all directories named src
find . -name src -type d
# Find all python files that have a folder named test in their
find . -path '*/test/*.py' -type f
# Find all files modified in the last day
find . -mtime -1
# Find all zip files with size in range 500k to 10M
find . -size +500k -size -10M -name '*.tar.gz'
```

grep - command generic tool for matching patterns from the input text

Finding shell commands
history | grep find

shell/environment/context

Environment - overall system config and settings that influence how software programs behave
ex
Environment variables - $PATH - specify the directories for executable files
Shell environment - aliases, functions and history
System settings - locale settings, network configurations
Context - specific conditions or circumstances in which something occurs or exists including dynamic runtime factors
ex
Script execution context - varies on current directory, user permissions, input
Program runtime context - system loads, user interactions, active processes
System state context - current state affecting operators (network availability, hardware status)

(..) - creates a new subshell/subenvironment

Process

An instance of computer program that is being executed, a fundamental unit of execution in multitasking operating system

Script as a process -
When you start shell it creates a process, when you spawn a script it creates new process, a process that is launched by another process is called subprocess

subprocess creation -
creates a subprocess with its own environment and context within the parent shell

Environment inheritance -
inherits parent shell' environment variables settings and current directory

Isolated changes -
changes in the current script are local unless exported to affect the parent shell

Process substitution, Command substitution

Process substitution - allows the output of a command to be used as if it were a file or stream; <(command), >(command)
For example
wc -l < <(sort myfile | uniq)
In practice you use a substituted command to another command
command substitution - replace command with what it returns; $()
myvar=$(command file)

Line endings

unix/linux - LF(\n)
Windows - CRLF(\r\n)
Mac - CR(\r)

Mounting

attribute(n) - quality assigned to
file - a container storing information
filesystem - method and data structure that operating system uses to manage files
Mounting - in unix/linux systems making a filesystem accessible at a certain point in the directory tree
ex in git
gitmodule is a repository
gitsubmodules is a mounted repository inside another
or a project above is called superproject

Path, export

PATH - special environment variable that contains colon separated list of directories
: - delimiter used in PATH to separate different directories
export - Command used to set environment variables be available to any subprocess started from the shell
$PATH - this references the current value of PATH

Example of appending a new path to PATH
mkdir -p $HOME/scripts
export PATH="$PATH:~/scripts" explain this lines bash
token -

Aliases, buffer, source etc

buffer - the memory referred to
When I start an interactive session, bash reads .bashrc for setting up environment variables, aliases, etc i.e customization of shell behaviour
source - reads and executes commands from a file into the current shell session environment, alias to source is a literal dot
command - might be an alias, export, ls, grep, or a shell function defined within a file

Homework 2

Wildcard - comes from card games where this represent any card, in computing this can be used to substitute for any other character or character
in unix:
* - any character
? - one character
.sh - shell scripting files format
POSIX - portable operating system interface to maintain compatibility between operating systems

File descriptor

A unique non negative integer that uniquely identifies an open file in a unix like operating system, is used to manage manage opened files and i/o streams
Standard descriptors
0 stdin

1 stdout

2 stderr

3 LECTURE EDITORS VIM

vim - modal editor, it switches between modes

Tab
Windows/Views - represents a buffer
esc - to exit any mode
: - command mode
Examples

- `:q` quit (close window)
- `:w` save (“write”)
- `:wq` save and quit
- `:e {name of file}` open file for editing
- `:ls` show open buffers
- `:help {topic}` open help
  - `:help :w` opens help for the `:w` command
  - `:help w` opens help for the `w` movement

4 DATA WRANGLING

Wrangling - handling managing controlling something, data in one format transforming to other format
Etymology old english - wringan - to twist
Piping is essentially data wrangling

Wildcard,capture groups,\b\B ()

wildcard - from card games where joker can be represented as any card
Capture groups - any info that can be referenced later
\b - word boundary (граница слова)
\B - not word boundary, match any position
() - group things together

Ex2 mistakes

I understood the exercise but did not inform myself enough by reading documentation to commands and to regex in general

5 CLI

lookup words that match this regex pattern
grep -i -E '([^aA]*[aA]){3}' words | grep -v 's$'

symlink

Symlink - greek symbolum i.e sign(like top of iceberg that refers to something) or token, something that represent something else,a point or reference to another file or directory, so when you access it os redirects you to the target file

Mean, median

mean - average sum of objects/number of objects
median - ascending sorted list, if the list count is odd, the middle number is the median, if the list count is even then the average of two middle numbers is the median
Hack: try to count them as indexes

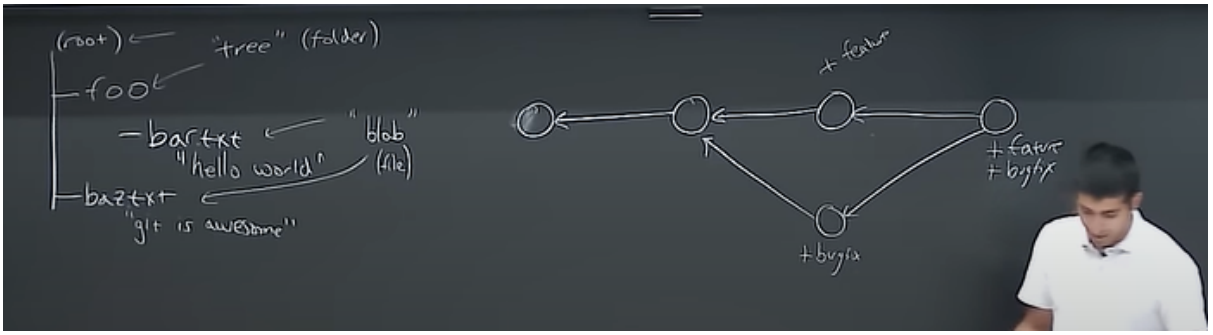
In awk
NF - number of fields
NR - number of

5 COMMAND LINE ENVIRONMENT
dotfiles - config file(I suspect this is data directed programming)
dotfiles repos - https://github.com/search?o=desc&q=dotfiles&s=stars&type=Repositories

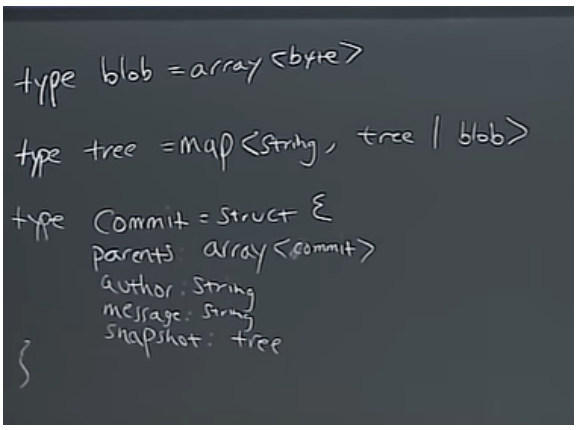
6 GIT, VCS

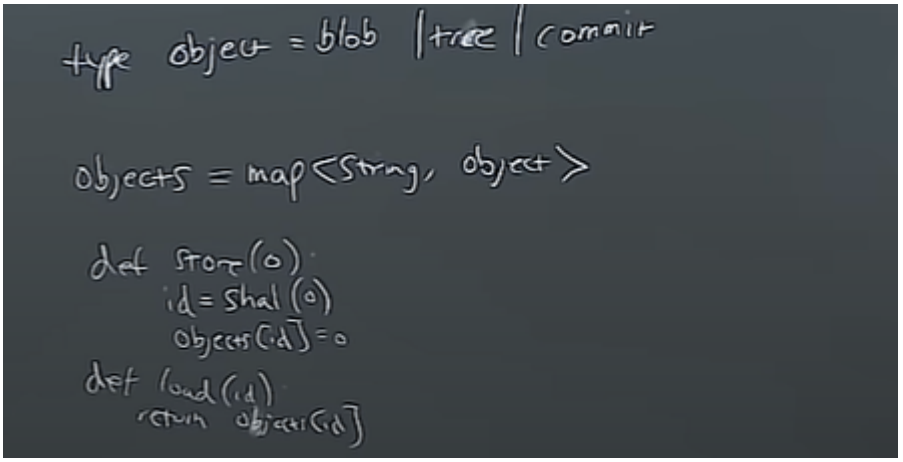
In general git is different from other vcs that it takes a snapshot of the change rather then delta, which implies line by line difference

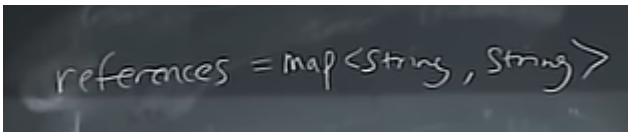
folder model - forest of trees
history model - linear but might be parallel



types i.e ADT







Objects are addressed by their hash
To address has you use reference

Git commit msg convention

Model
Capitalized, short (50 chars or less) summary
why in total max 72 chars

Let's start with a few of the reasons why wrapping your commit messages to 72 columns is a good thing.

- `git log` doesn't do any special special wrapping of the commit messages. With the default pager of `less -S`, this means your paragraphs flow far off the edge of the screen, making them difficult to read. On an 80 column terminal, if we subtract 4 columns for the indent on the left and 4 more for symmetry on the right, we're left with 72 columns.
- `git format-patch --stdout` converts a series of commits to a series of emails, using the messages for the message body. Good email netiquette dictates we wrap our plain text emails such that there's room for a few levels of nested reply indicators without overflow in an 80 column terminal. (The current rails.git workflow doesn't include email, but who knows what the future will bring.)

Example commands
reset current stage
git reset HEAD

tags - 1.0.0 - major.minor.patch

7 DEBUGGING PROFILING

Profiling - outline the performance of a program
Remark on thinking framework -> Focus on why before how; look at general, then in isolation, then how they relate to each other

Scientific units - 1e-09 = 1x10^-9

8 METAPROGRAMMING

Make -
A tool to automatically recompile and manage dependencies
makefile - config file
phony targets - no refer to file, only action
Continuous integration - a property of this: event driven
Hook functions - callback that allows to extend or modify some code during runtime

shopt - shell options, flag -s set, flag -u undo, globstar - match all files including all directories and subdirectories
.lockfiles - the exact version you are currently depending on for each dependency

9 SECURITY CRYPTO

Entropy - etymology: inside trope; measured in bits, measure randomness, higher the entropy, more disorder and harder to choose
For example concatenated helloprettykittysuity could have has higher entropy than Tr0ub4dor&3 if each word would have been chosen from a set of 2000 words