**Introduction**

Unix: operating system and set of tools

command line == shell == console

Bash is default shell program on Mac and Ubuntu

**Command structure**

**command options arguments 🡪** options proceed “-“

**Directory structure**

**Absolute path**: exact location within the computer which starts at “/” known as the **root**

**Relative path**: relative location to current directory

**cd** 🡪 change directory

.. 🡪 parent . 🡪current ~🡪home / 🡪 root

* Tip: you can push tab to autofill the names
* Tip: accessing drives cd /mnt/c

**Data and Directories**

**Data**🡪 file, **directory**🡪folder

**pwd** 🡪 print working directory

**ls** 🡪 list of available files and folders

**ls \*2017** 🡪 wild card all files end with 2017

**ls -R 🡪** all directories and their directories

**ls -R -F 🡪** adds a “\*” after each executable

**ls -l** 🡪 detailed info and permission of files and dir (d)

**mkdir**🡪 making a folder

**cut -f 2-5,8 -d , data.csv** 🡪 getting columns from a csv file, f(filed), 2-5,8(colums), -d (delim), ,(comma)

**Migration and destruction**

**cp file\_copy file\_paste 🡪** getting a copy in the same dir

**cp file folder** 🡪copying a file to another dir

**cp -r folder folder** 🡪 recursive copy (move all content)

**mv source\_file destination** 🡪 move file to dest

**mv file\_old file\_new** 🡪 renaming files

**mv file1 file2 destination**🡪 moving several files

**mv folder1 folder2** 🡪moves folder1 inside folder2

**rm file** 🡪 removes the file

**rm -r directory** 🡪removing directory

**rmdir**🡪 removing folder

**Getting help**

**man command** 🡪 getting help on the command

**apropos function 🡪** find all the commands that does something with function

**Verifying the content**

**cat file 🡪** viewing the content in the command line

**cat file1 file2** 🡪 concatenates them

**less file** 🡪 for larger files, you can go inside and go back and forth inside it using : p , :n

**head -num files** 🡪 gives n higher lines

**head file** 🡪 gives the 10 line headers

**tail file** 🡪 exactly the same as head just from bottom

**wc file** 🡪 gives word count of a file (lines, words, char)

**touch filename** 🡪 creates a new file

**nano file** 🡪 creating/opening a file to edit

**Writing outputs**

**echo “hi” 🡪** writes hi on the shell

**echo “hi” >file 🡪** create a new file and print hi in it

**echo “hi” >> file 🡪** append hi to the end of the file

**Regular expressions**

**grep ‘re’ string 🡪** looks for re patterns in string

**egrep ‘re’ string** 🡪 same as grep dealing with methachars

**egrep -n ‘re’ string** 🡪gives the line number of matches

**RegEx refresher**:

+🡪one or more, \*🡪 zero or more, {n}🡪 exact n times, **^🡪**complement of expression, (group)🡪capturing group, \w🡪all words, \d🡪all numbers, \s🡪space, |🡪or

**Accessories**

**~/.bash\_history 🡪** history of the commands

~/.bash\_profile 🡪 runs on start (use foe alas creation)

**alias sth = ‘some command’🡪** alias creation

**source ~/.bash\_profile**🡪 activates the bash profile

**diff file1 file2** 🡪 shows different lines in a file

**sdiff file1 file 2**🡪 shows differences side by side

**md5** 🡪 generates the hash of the file

**|**🡪pipe operator: takes the output of one command and use it as the input to the next

**Math, Variables, and Functions**

**expr 5 + 5🡪** evaluate the math and returns 10 c

**\\*🡪**mult**, +->**sum**, /🡪** int div**, % 🡪**mod

**var=5🡪** variable definition without white space

**echo $var🡪** when wat to echo variable put $ behind it

**let var=$var+1**🡪 to modify variables

**var2=$(cat txt)🡪** getting a result of command in a variable, also called command substitution $,()

**var3=”hi $var2 !”** 🡪 including variables in strings

**$@** 🡪 all the arguments put in function

**$1**🡪 first argument put in function

**$#** 🡪 number of arguments put in function

**read input** 🡪 reading the IO into input variable

**Conditional**

**exit 0 🡪**exiting without error

**exit 1 🡪**exit with error

**true && echo “hi”** 🡪 run right program (conditional exe)

**[[]]** 🡪 conditional expressions

**[[ 4 -gt 3 ]]**🡪 true since 4>3

**[[ -e file.txt ]]** 🡪 true when file exist

**gt, ge, lt, le,e** 🡪 comparison flags

**string =~ regex pattern** 🡪 conditional to check reg match

**[[ ! ]]** 🡪 Not on the rest of statement in bracket

**=, !=** 🡪 string equal to, string not equal to

**if [[]]**

**then**

**do something**

**elif**

**do something else**

**else**

**do last one**

**fi** 🡪 if statement (indentations are not required)

**Arrays**

**list=(A B C)**🡪 creating an array

**${list[0]}🡪** retrieving arrays

**${list[\*]}🡪** retrieving all the element

**list[4]=D** 🡪 separately assigning each element

**echo ${list[\*] :5:3}** 🡪 all from index 5, how many=3

**#list[\*]** 🡪 length of array

**list+=(a b c)**🡪 appending to the end of a list

**Braces**

**{from .. to} 🡪** generates a sequence from “from” to “to”

**{1..10} 🡪** 1,2,3,4,4,5,6,7,8,9,10

**{a..d} 🡪** a,b,c,d

**{a..c}2 🡪** a2, b2, c2

**{1..3}{A..C} 🡪** 1A, 1B, 1C,..,3C

**eval echo {$start .. $end} 🡪** to sequence on variables start, end

**Iterations**

**for i in {1..3}**

**do something**

**done 🡪** structure of a for loop

**for files in $(ls) 🡪** iterates over all the file names in ls

**while [[condition]]**

**do something**

**done 🡪** structure of a while loop

**Function**

**function [name] {**

**}**🡪 function declaration

**$1, $2, $3, $#, $@** 🡪 reading arguments inside the function

**source script.sh** 🡪 to access the functions defined inside the script.sh

**local val=0**🡪 defining local variables inside functions. Good practice since the variables are global.

**Unix Programs**

Their characters:

* Limit to do one thing well
* Short program
* Practice pipelining

**permission r** 🡪 read

**permission w** 🡪edit

**permission x** 🡪execute

**chmod** 🡪 changes permission of the files

**chmod level action permission** 🡪 chmod structure

**u,g,o,a** 🡪 chmod permission levels

**+, -, =** 🡪 actions (add permission, remove permission, set permission)

**r,w,x** 🡪 permissions (read, edit, execute)

**chmod u+x script.sh** 🡪 add execution permission for script.sh for anyone

**./executable.sh** 🡪 how to run the executables

**#!** 🡪 SHEBANG: located at the beginning of program to let user know how to run the program

**#!/usr/bin/env bash** 🡪 running the program with bash

**Environmental Vari****ables**

Provides info on your current computing environment.

**#HOME** 🡪 location of home directory

**#PWD** 🡪 current directory

**$PATH**🡪 sequence of path separated by column. Shell looks there for commands

**Communication**

**curl -o** [**https://......csv**](https://......csv) 🡪 downloading from the internet

**API**🡪 set of rules which allows you to communicate with a we server or programs. You can curl from APIs with different arguments.