A Gentle Introduction To Linux OS, Bash Scripting and Python Programming Language

December 1, 2022

OBI I.A.

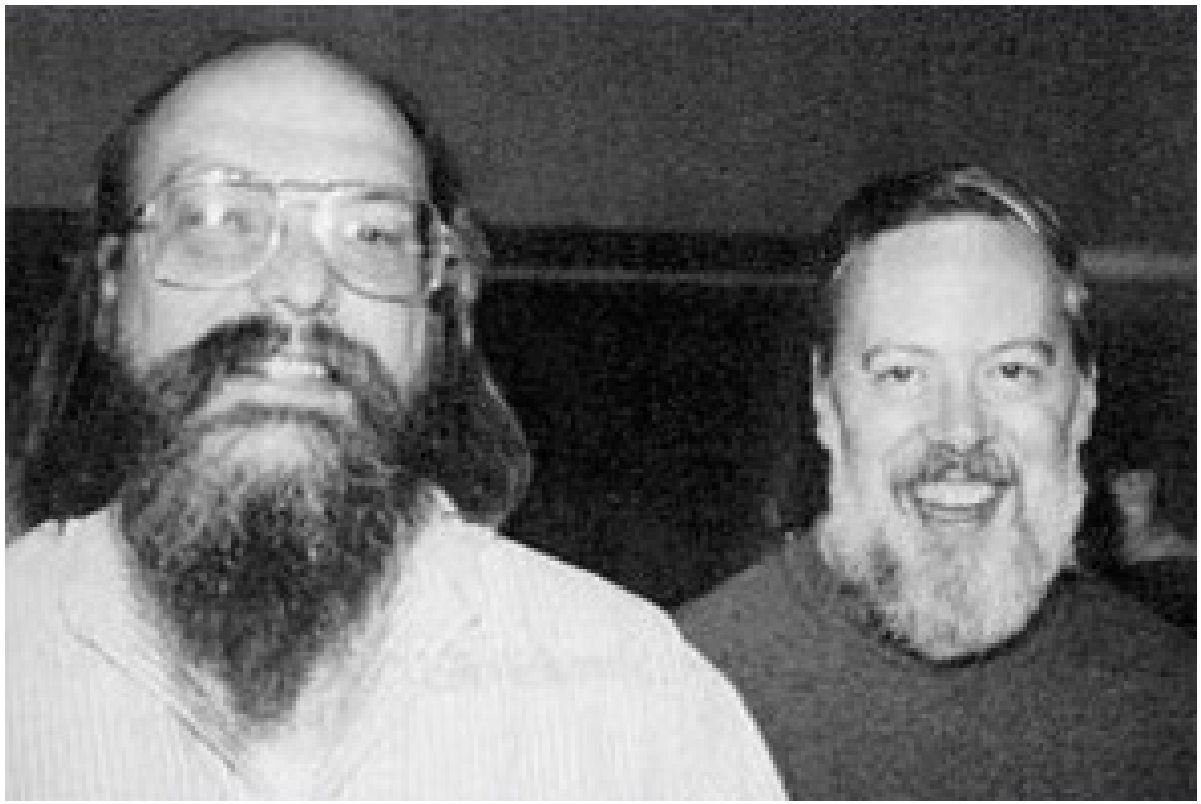
Hands-On Training Workshop on Radio Astronomy for University Lecturers

Outline

- Introduction and History of the Linux OS
- The Linux Operating Systems (OS)

Short History of Linux OS

- 1969 Unix operating system was conceived and implemented by Ken Thompson and Dennis Ritchie of AT & T



Short History of Linux OS

- 1983 Richard Stallman started the GNU project with the goal of creating a free UNIX-like operating system

Short History of Linux OS

- 1986 Maurice J. Bach, of ATT Bell Labs, published The Design of the UNIX Operating System.

Short History of Linux OS

- 1986 Maurice J. Bach, of ATT Bell Labs, published The Design of the UNIX Operating System.
- 1987 MINIX, a commercial Unix-like system was released

Short History of Linux OS

- 1986 Maurice J. Bach, of ATT Bell Labs, published The Design of the UNIX Operating System.
- 1987 MINIX, a commercial Unix-like system was released
- 1991 Linus Torvalds at the age of 25 cloned MINIX

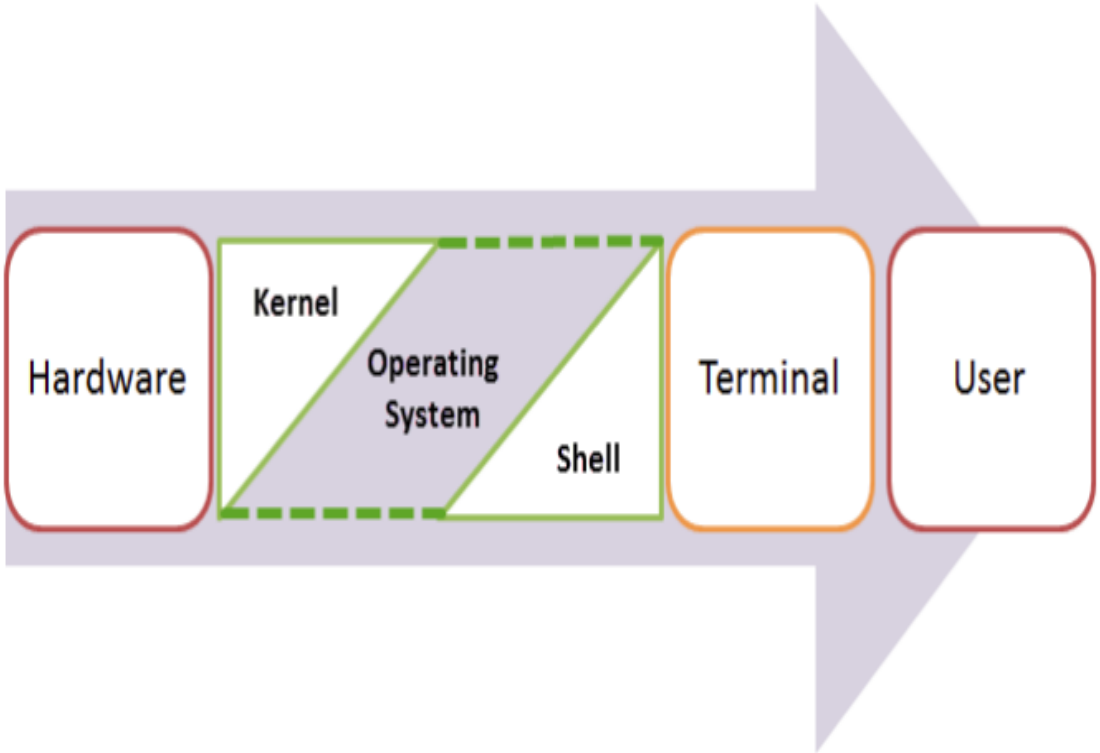
Other remarkable achievements of AT & T Company

- Alexander Graham Bell credited with inventing and patenting the first practical telephone co-founded AT & T
- The charge-coupled device (CCD) was invented by Willard Boyle and George E. Smith in 1969

Other remarkable achievements of AT & T Company

- Alexander Graham Bell credited with inventing and patenting the first practical telephone co-founded AT & T
- The charge-coupled device (CCD) was invented by Willard Boyle and George E. Smith in 1969
- The discovery of the first radio waves from the Milky way by Karl Jansky in 1931 gave birth to the field of Radio Astronomy

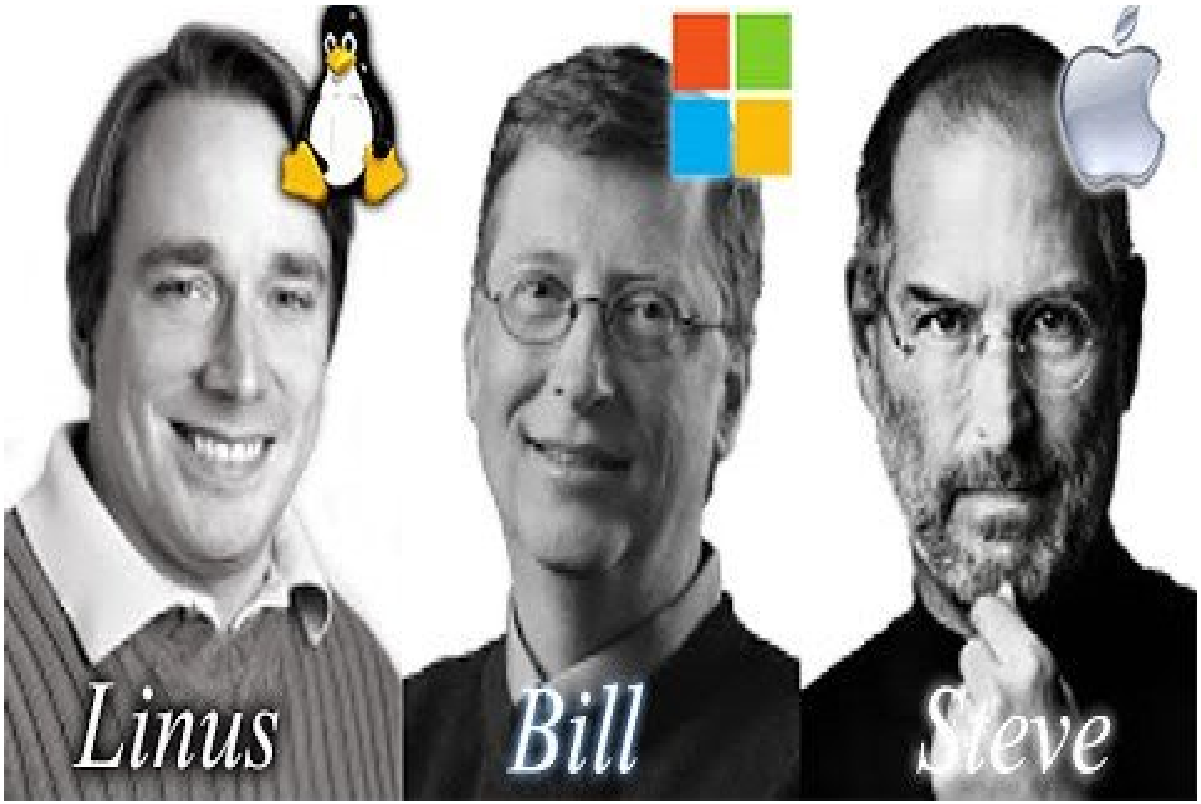
Structure of the Unix Operating System



Structure of the Unix Operating System

KERNEL	SHELL
A computer program which acts as the core of the computer's operating system and has the control over everything in the system	A computer program which works as the interface to access the services provided by the operating system
Core of the system that controls all the tasks of the system	Interface between the kernel and user
Does not have types	Has types such as Bourne shell, C shell, Korn Shell, Bourne Again Shell, etc.

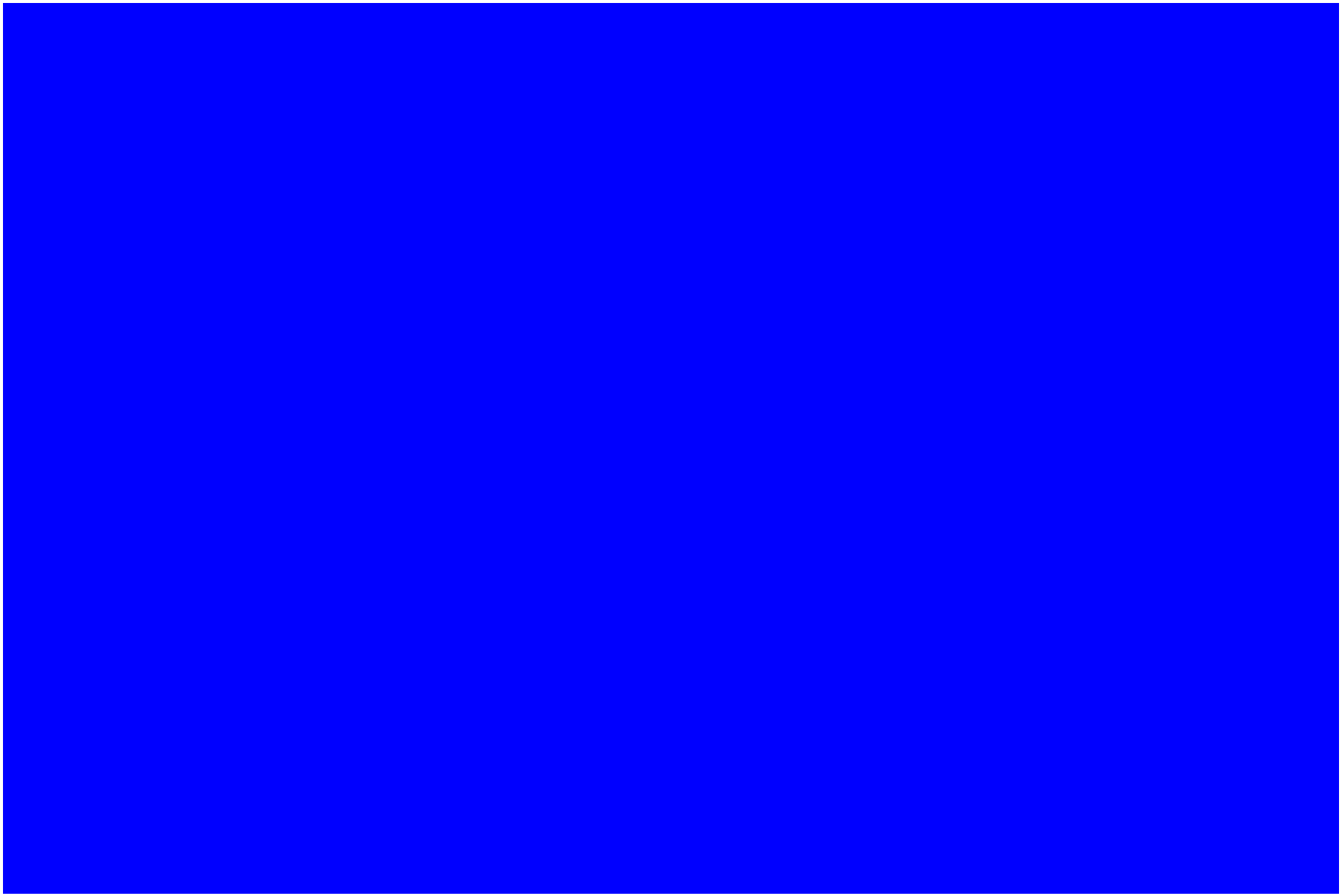
Other OSes (Unix-Based and Non-Unix-Based)



Who Do You Think Has A Bigger Influence?

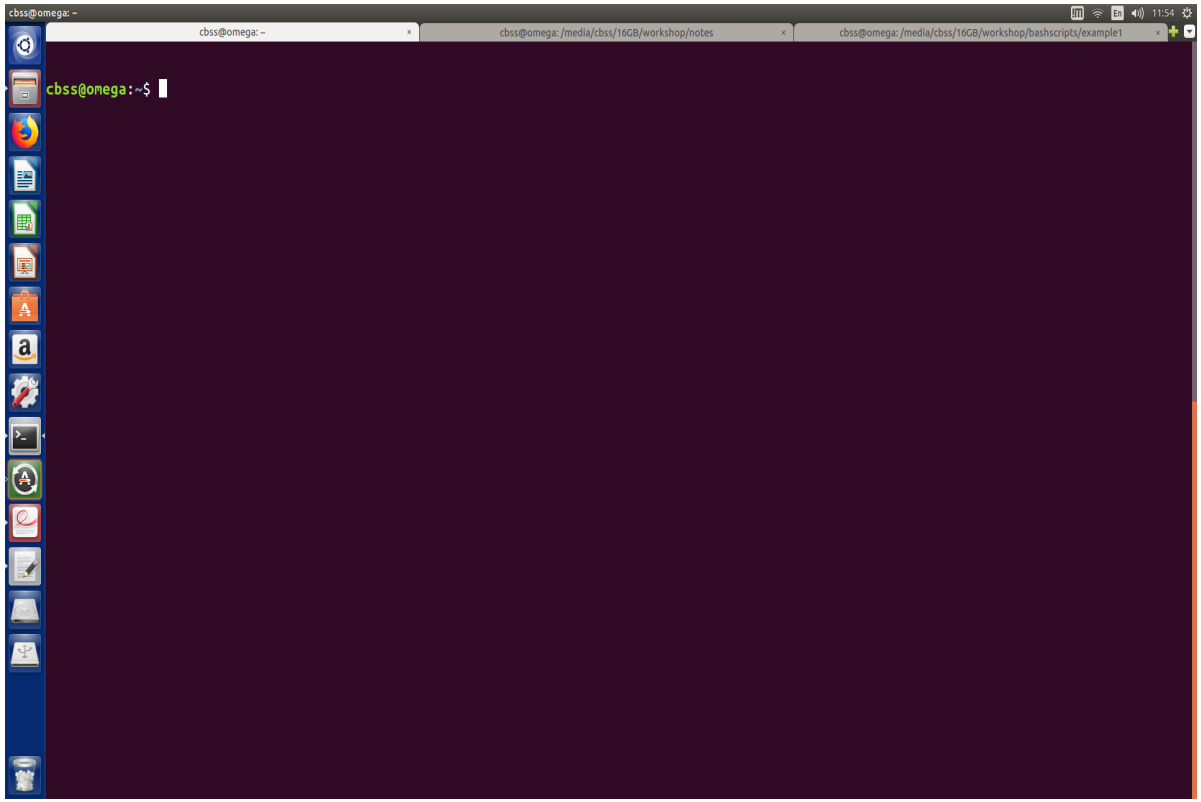
A computer is like air conditioning system, it becomes useless when you open Windows

Linus Trovalds



The Linux OS - The Command Line Interface (CLI)

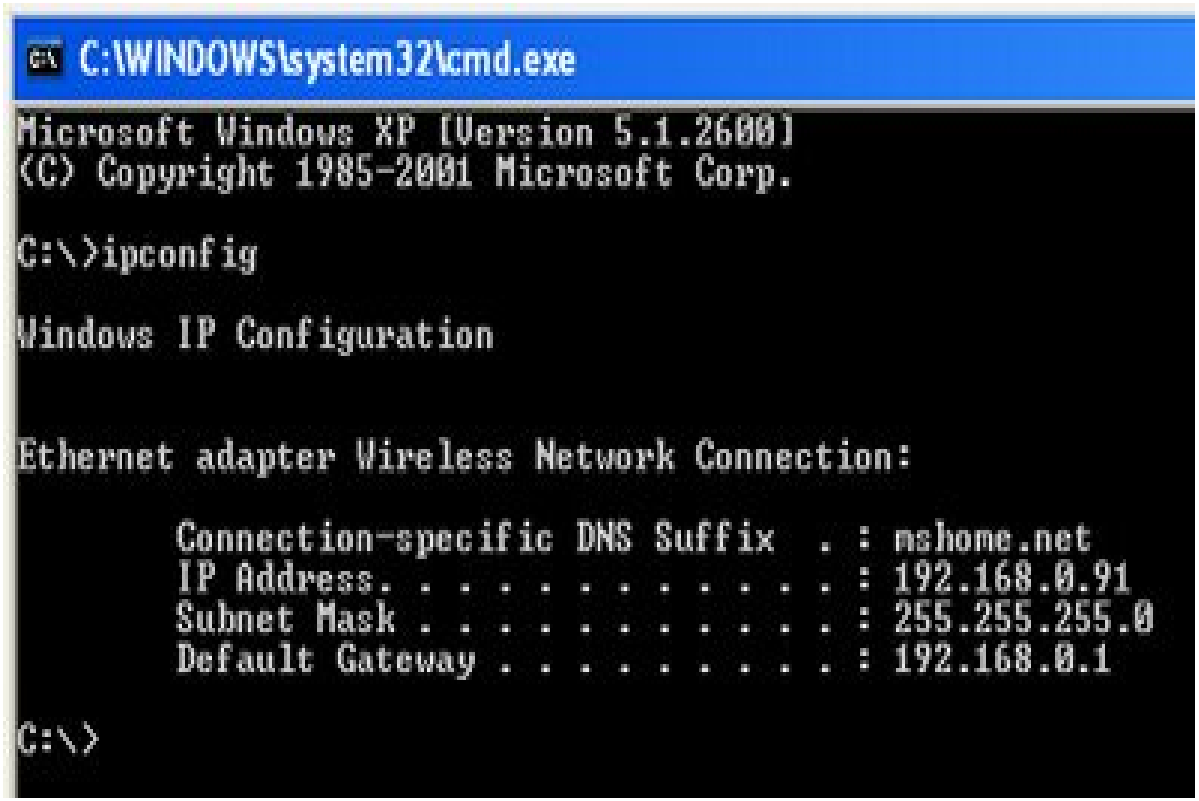
- A powerful and interesting beast, don't worry, a bit of practice will make it you friend
- This workshop will focus on the Ubuntu CLI rather than GUI



The Linux OS - Does Windows have a CLI?

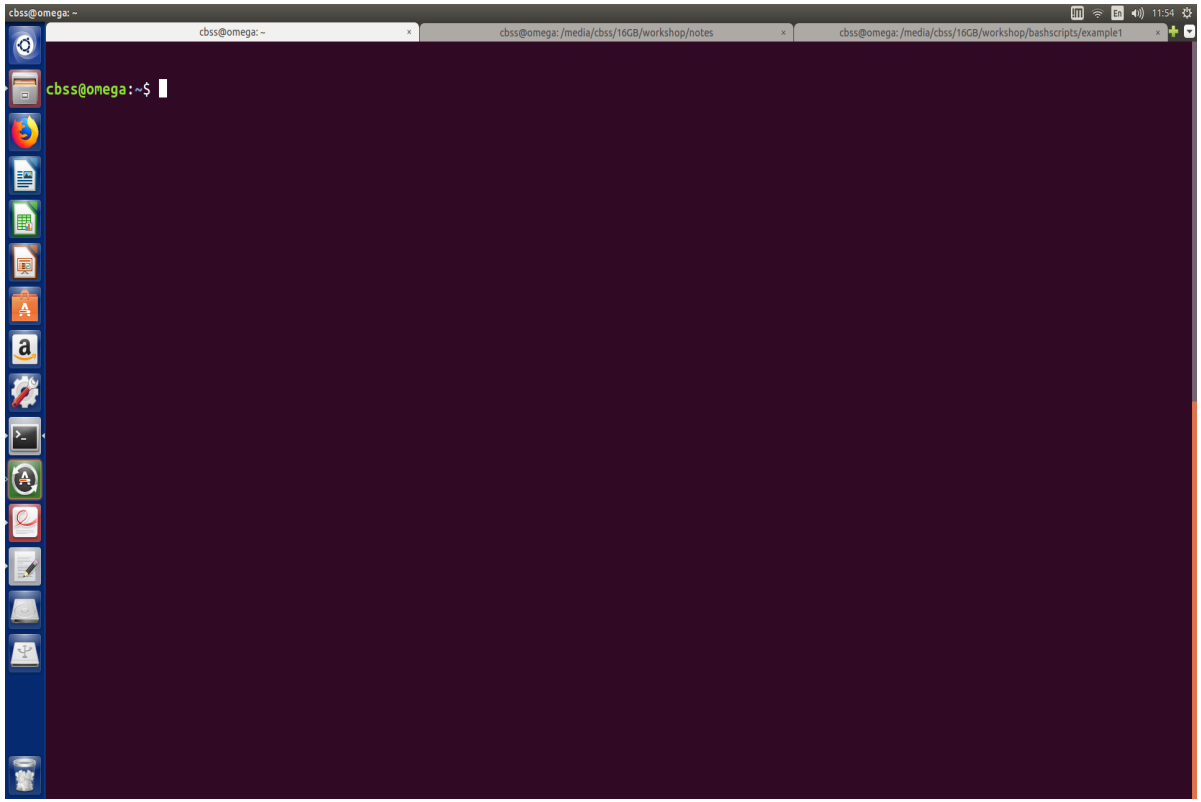
The Linux OS - Does Windows have a CLI?

- Yes, referred to as MS-DOS though very weak



The Linux OS - The Command Line Interface (CLI)

- A powerful and interesting beast, don't worry, a bit of practice will make it you friend
- This workshop will focus on the Ubuntu CLI rather than GUI



The Linux OS - Basic Navigation

- We'll learn how to move around the system and this forms the foundation of being able to work effectively in Linux
- So where are we now?

The Linux OS - Basic Navigation

- We'll learn how to move around the system and this forms the foundation of being able to work effectively in Linux
- So where are we now?
  $ pwd (short for 'print working directory')
- Ok, now i know where, what is there?

The Linux OS - Basic Navigation

- We'll learn how to move around the system and this forms the foundation of being able to work effectively in Linux
- So where are we now?
  $ pwd (short for 'print working directory')
- Ok, now i know where, what is there?
  $ ls (short for 'list')
- How do I get to somewhere else?

The Linux OS - Basic Navigation

- We'll learn how to move around the system and this forms the foundation of being able to work effectively in Linux
- So where are we now?
  $ pwd (short for 'print working directory')
- Ok, now i know where, what is there?
  $ ls (short for 'list')
- How do I get to somewhere else?
  $ cd (short for 'change directory')

The Linux OS - Basic Navigation

- Believe it or not, Everything In Linux Is A File
- A text file is a file, a directory is a file, your keyboard is a file
- Keeping this mind is a secret to understanding Linux behaviour
- A file extension is a set of 2 - 4 characters after a full stop at the end of a file (e.g. participants.txt, whatsapp.exe, picture.jpg)
- Linux is Case Sensitive unlike Windows
  e.g. participants.txt, Participants.txt and particiPantS.txt
  are all different files in Linux but same file in Windows

The Linux OS - Man Pages

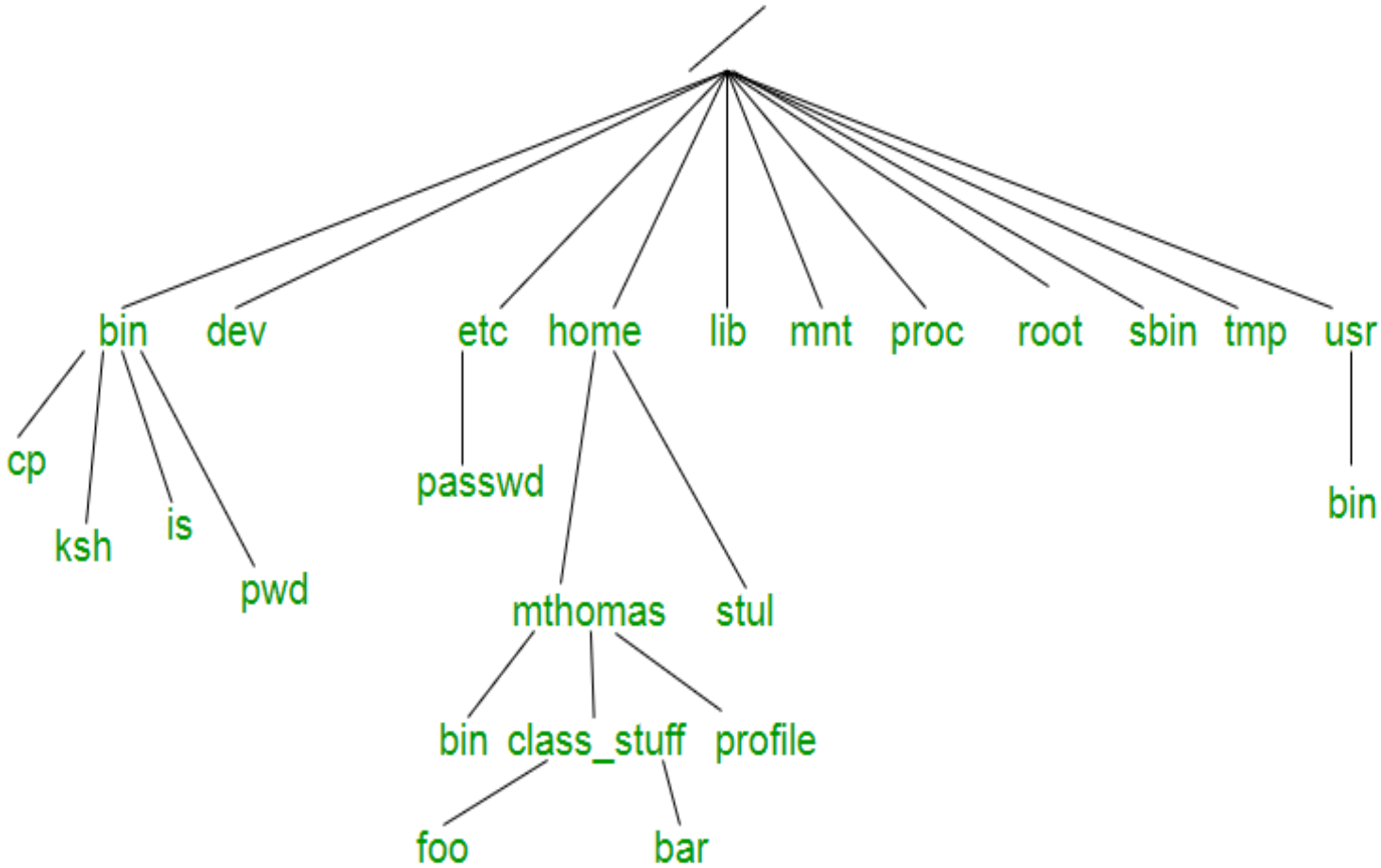
- What are they exactly?

The Linux OS - Man Pages

- What are they exactly?
- The manual pages are a set of pages that explain every command available on your system including what they do, the specifics of how you run them and what command line arguments they accept
  eg. **$ man ls** (manual page for ls, type 'q' to quit)

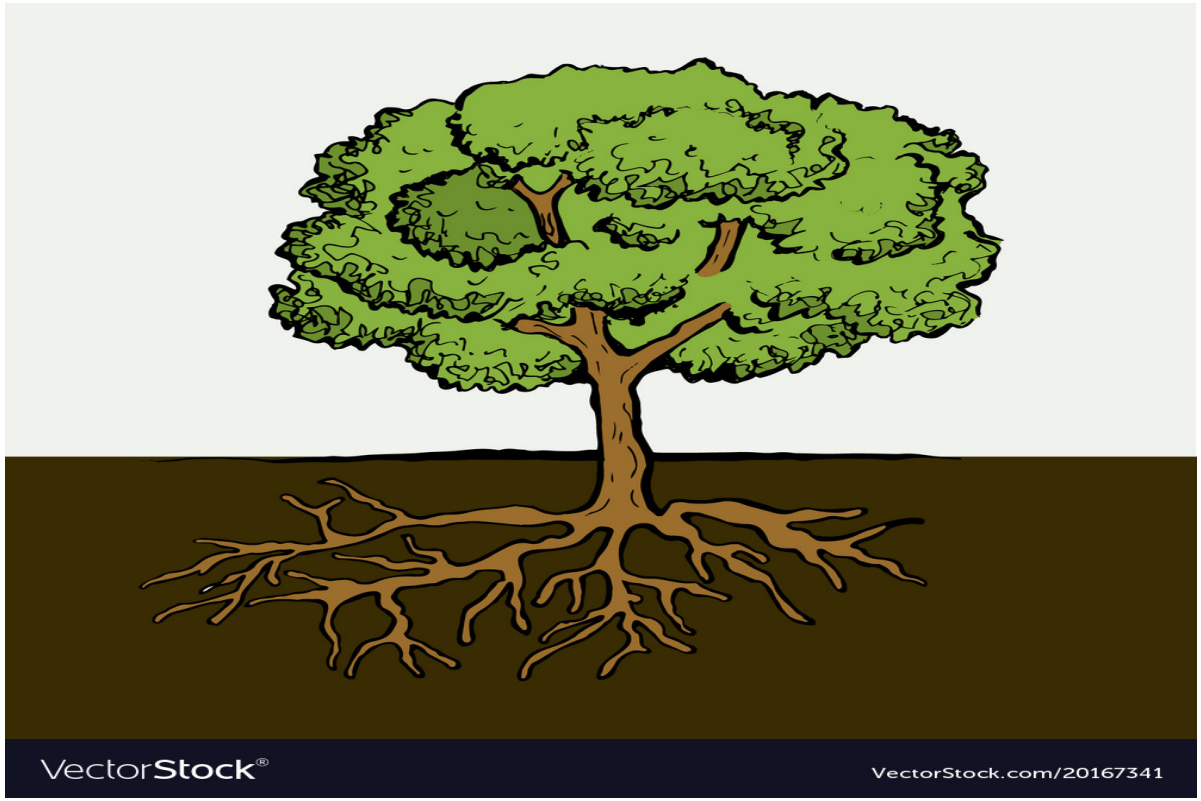
The Linux OS - File System and File Manipulation

- Linux organises it's file system in a hierarchical way



The Linux OS - File System and File Manipulation

- Real Life Analogy



The Linux OS - File System and File Manipulation

- How do I manipulate a file

The Linux OS - File System and File Manipulation

- How do I manipulate a file
- I mean how do i create, rename(try), move and delete a file?
  $ touch, cp, mv and **rm** respt. (shorts for 'copy, move and remove')

The Linux OS - File System and File Manipulation

- How do I manipulate a file
- I mean how do i create, rename(try), move and delete a file?
  $ touch, cp, mv and **rm** respt. (shorts for 'copy, move and remove')
- What about a directory, how do I create, rename, move and delete a directory?

The Linux OS - File System and File Manipulation

- How do I manipulate a file
- I mean how do i create, rename(try), move and delete a file?
  $ touch, cp, mv and **rm** respt. (shorts for 'copy, move and remove')
- What about a directory, how do I create, rename, move and delete a directory?
  $ mkdir, cp, mv and **rmdir** respt. (shorts for 'makedirectory, copy, move and remove directory')

The Linux OS - File System and File Manipulation

- How do I manipulate a file
- I mean how do i create, rename(try), move and delete a file?
  $ touch, cp, mv and **rm** respt. (shorts for 'copy, move and remove')
- What about a directory, how do I create, rename, move and delete a directory?
  $ mkdir, cp, mv and **rmdir** respt. (shorts for 'makedirectory, copy, move and remove directory')
- Have tried out the above commands, Can you recall any other kind of file manipulations?

The Linux OS - Text Editors

- We created empty files in the last section. How we fill it up with texts just like do in a Microsoft Words Notepad application?

The Linux OS - Text Editors

- We created empty files in the last section. How we fill it up with texts just like do in a Microsoft Words Notepad application?
- Text Editors are applications to put content into files and edit that content as well There are CLI (e.g. nano, vi, vim) and GUI (notepad, edit, textedit) text edittors

The Linux OS - WildCard

- Wildcards (* ?) are a set of building blocks that allow you to create a pattern defining a set of files or directories
- e.g **$ls -D*** (list all files/dirs beginning with D)

The Linux OS - Ownerships and Permissions

- Every file and directory on your Unix/Linux system is assigned 3 types of owner:
  user(u), group(g) and others(o)
- Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners
  read(r), write(w) and execute(x)

The Linux OS - Ownerships and Permissions

- Every file and directory on your Unix/Linux system is assigned 3 types of owner:
  user(u), group(g) and others(o)
- Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners
  read(r), write(w) and execute(x)
- What the heck do all these mean?

The Linux OS - Ownerships and Permissions

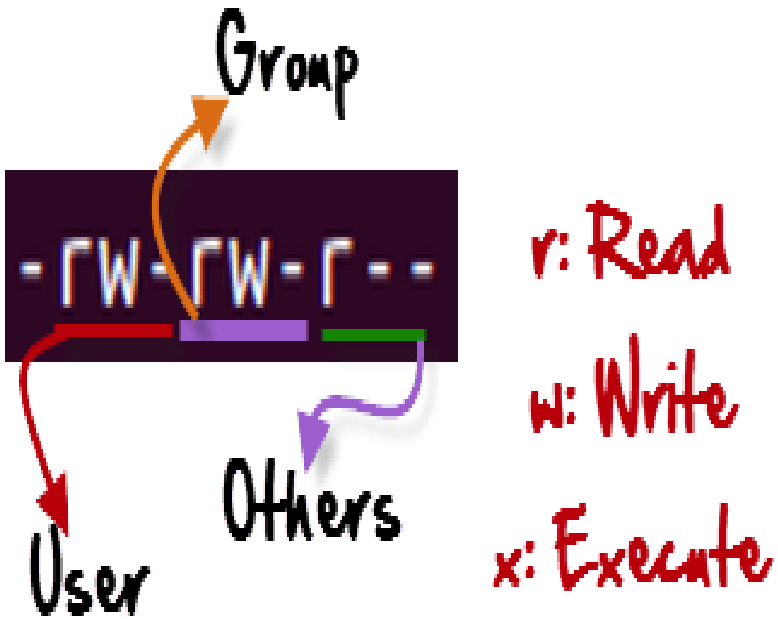
- Every file and directory on your Unix/Linux system is assigned 3 types of owner:
  user(u), group(g) and others(o)
- Every file and directory in your UNIX/Linux system has following 3 permissions defined for all the 3 owners
  read(r), write(w) and execute(x)
- What the heck do all these mean?
  - (r) - open and read a file
  - (w) - write permission of a dir enables u to rename, add/move file out of a directory
  - (x) - You recall '.exe' in windows? In Linux you can only execute (run) a program if and only if the program has the execute permission

The Linux OS - Ownerships and Permissions

- Sounds interesting, so how do I check these various 'r,w and x' permissions for a file?

The Linux OS - Ownerships and Permissions

- Sounds interesting, so how do I check these various 'r,w and x' permissions for a file?
- Very simple, just type **$ls -l filename**



The Linux OS - Ownerships and Permissions

Number	Permission Type	Symbol
0	No Permission	- - -
1	Execute	- - x
2	Write	- w -
3	Execute + Write	- w x
4	Read	r - -
5	Read + Execute	r - x
6	Read + Write	r w -
7	Read + Write + Execute	r w x

The Linux OS - Ownerships and Permissions

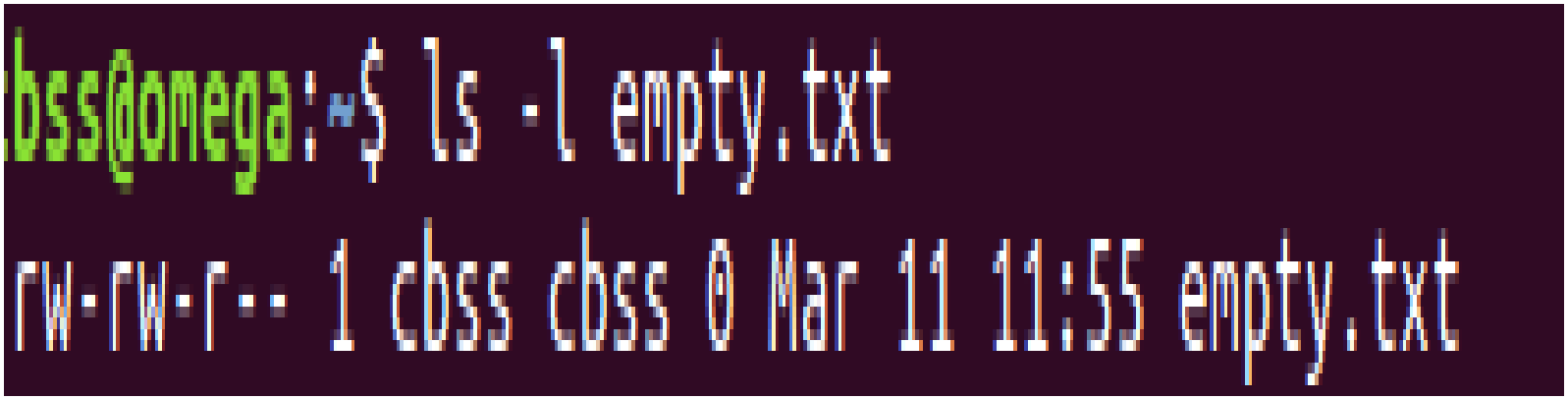
- Who among the three users can change a file's permission and how can it be done ?

The Linux OS - Ownerships and Permissions

- Who among the three users can change a file's permission and how can it be done ?
- Again very simple, the command **chmod** (short for change mode)will do so
  Syntax is chmod permission-type filename
- 'permission-type' above can be numerical (e.g. 754) or symbolic (eg. +x) value

The Linux OS - Ownerships and Permissions

- Let's try this out, i.e. changing permissions of a file (Using table on slide 25)



The Linux OS - Ownerships and Permissions

- Let's try this out, i.e. changing permissions of a file (Using table on slide 25)

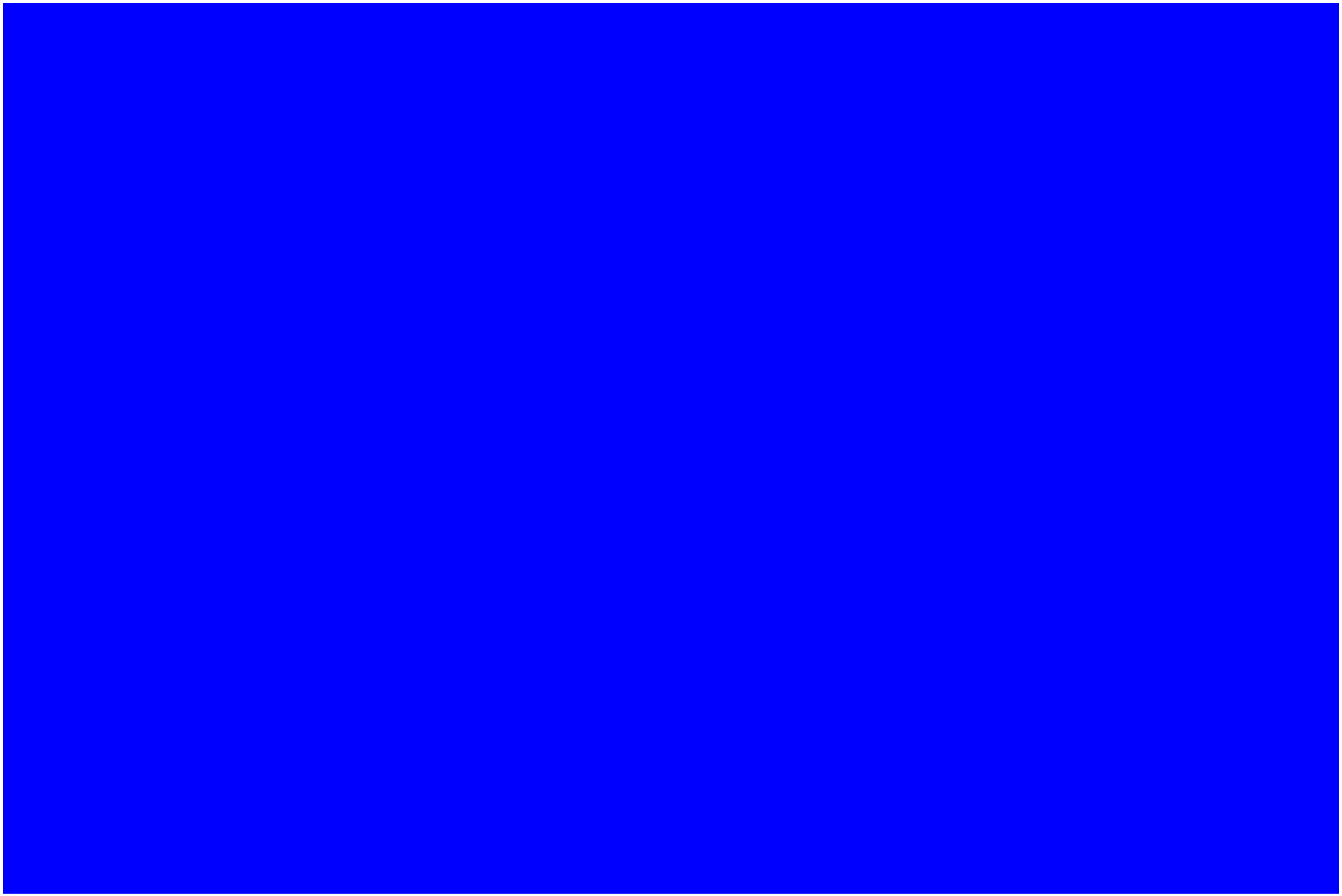
```
cbss@omega:~$ ls -l empty.txt
-rw-rw-r-- 1 cbss cbss 0 Mar 11 11:55 empty.txt
```

```
cbss@omega:~$ chmod 666  empty.txt
cbss@omega:~$ ls -l empty.txt
-rw-rw-rw- 1 cbss cbss 0 Mar 11 11:55 empty.txt
```

The Linux OS - Ownerships and Permissions

- And back to the default/previous permission .....

```
cbss@omega:~$ chmod 664  empty.txt
cbss@omega:~$ ls -l empty.txt
-rw-rw-r-- 1 cbss cbss 0 Mar 11 11:55 empty.txt
```



The Linux OS - Piping and redirection

- We'll try out examples given in Linux Commands document distributed

  $ cd

  $ ls ⟩ /home/cbss/workshop/examples/empty.txt

  $ ls ⟩⟩ /home/cbss/workshop/examples/empty.txt

The Linux OS - Process Management

- We'll try out examples given in Linux Commands document distributed
- open firefox web browser or any GUI application
  $ ps aux
- Try to kill this application that you've opened using the **kill** command and it's **PID**

The Linux OS - Filters

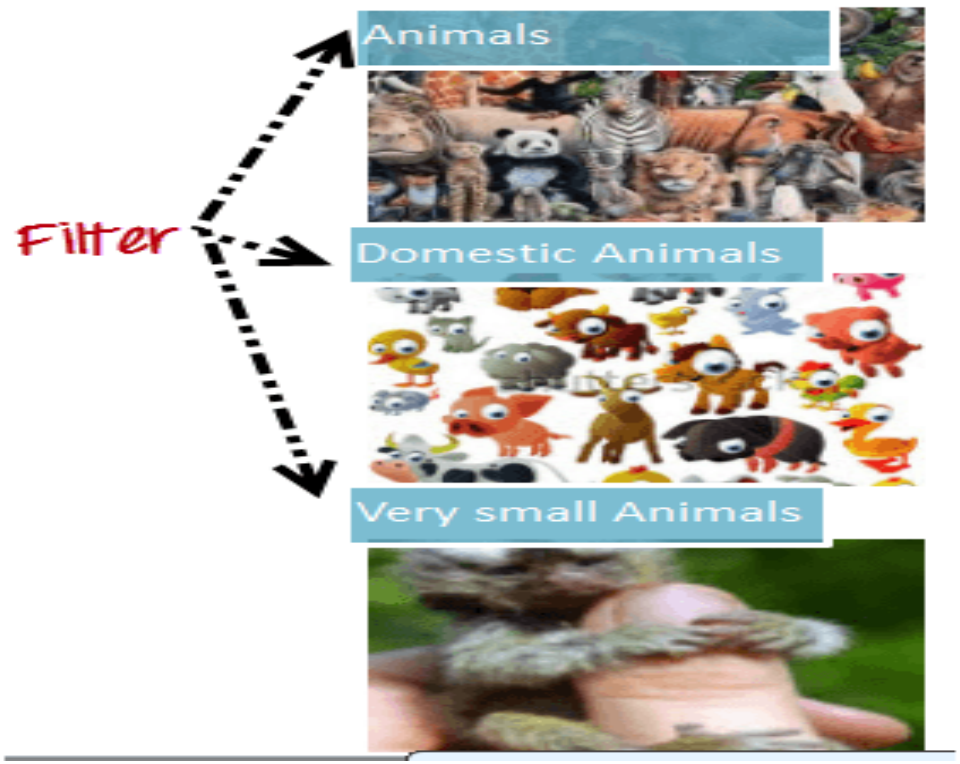
We'll try out examples given in Linux Commands document distributed

You all have a copy of the participant's list?

The Linux OS - Filters

We'll try out examples given in Linux Commands document distributed

You all have a copy of the participant's list?

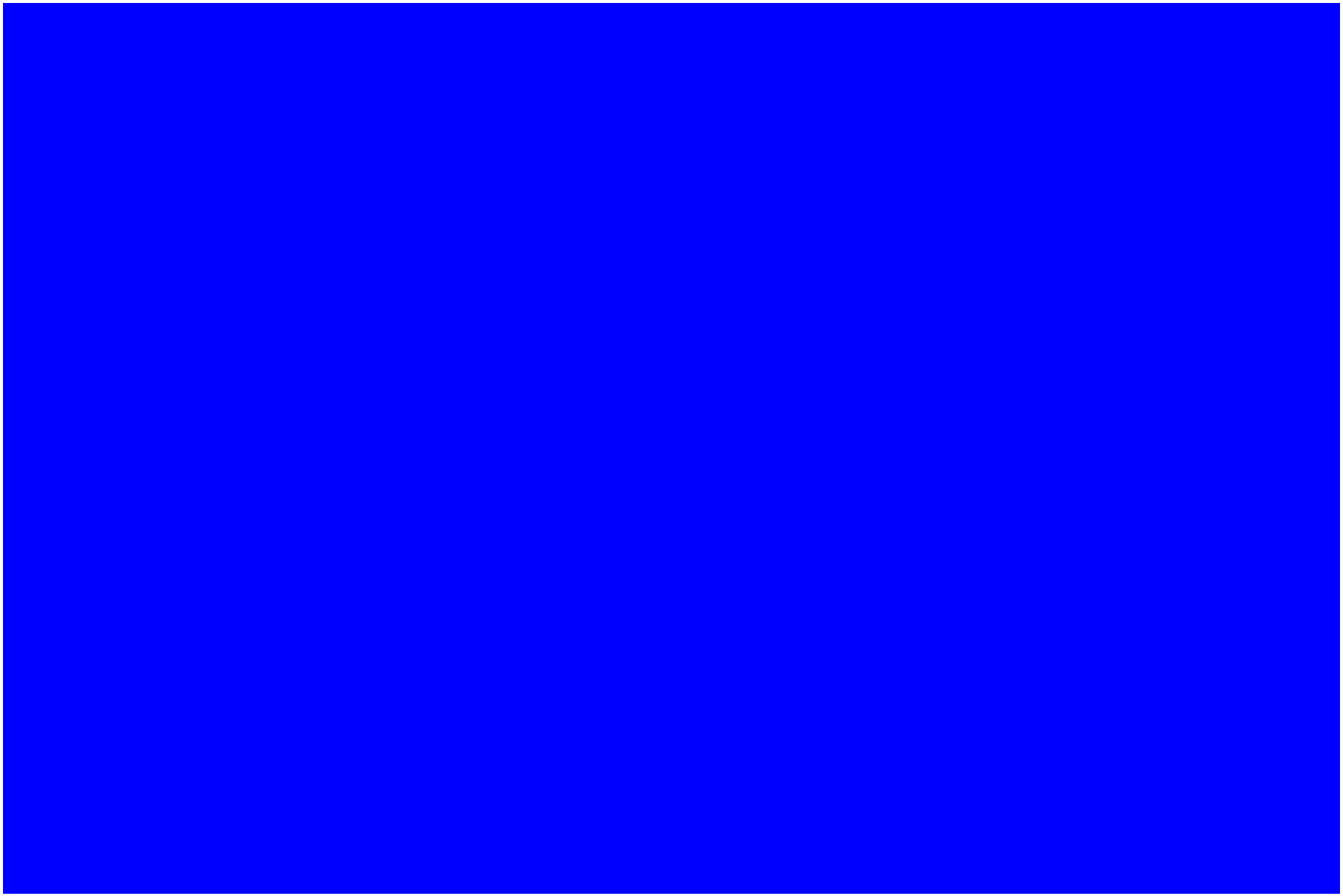


eg.

that we'll use in the next section?

The Linux OS - Linux Administration

- Linux is a multi-user OS, there is need of an administrator who can manage many user accounts, their rights (permissions) and the overall system security
- We'll try out examples given in Linux Commands document distributed



Bash Scripting- Why a Bash Script?

Bash Scripting- Why a Bash Script?

Intelligence is the ability to avoid doing work, yet getting the work done

Bash Scripting- Why a Bash Script?

Intelligence is the ability to avoid doing work, yet getting the work done

Good news for the very busy and lazy CBSS Science officers

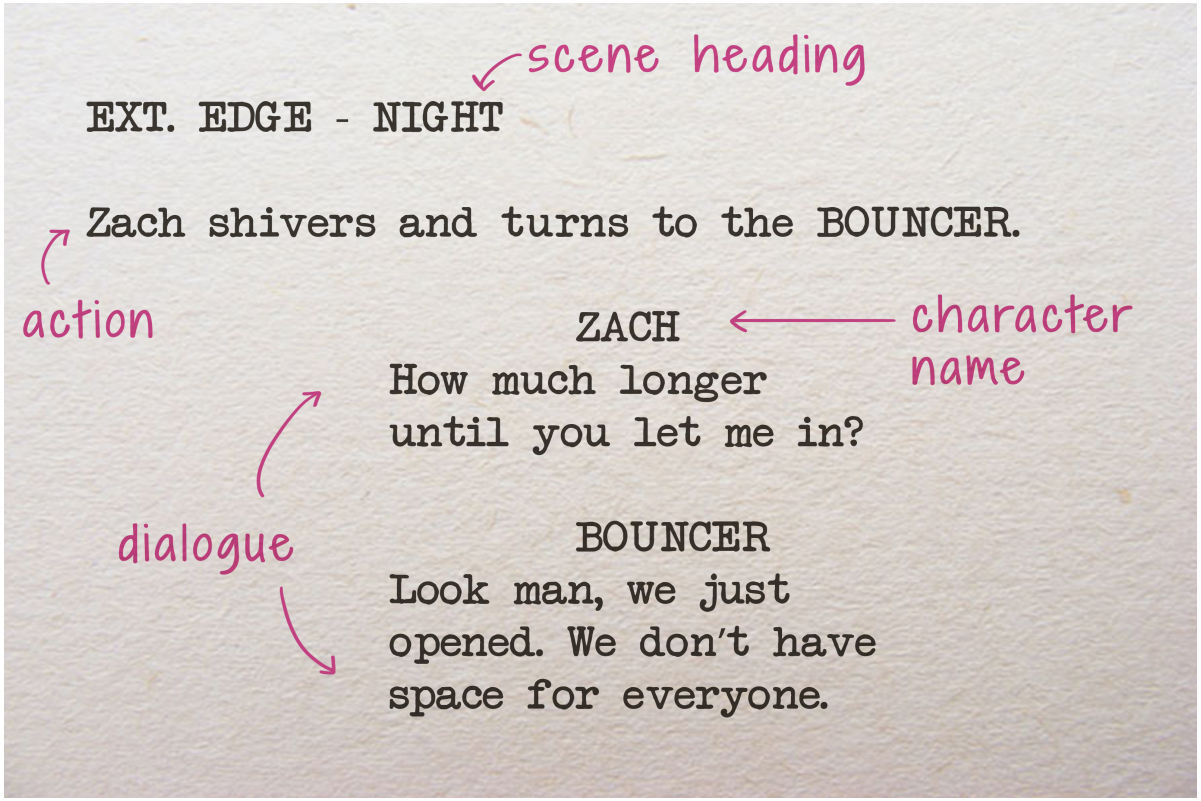
Bash Scripting - Why a Bash Script?

So what exactly is a bash script?

Bash Scripting - Why a Bash Script?

So what exactly is a bash script?

But wait, think of a film script



Bash Scripting - Why a Bash Script?

- A film script tells the actors what they should say and do
- A bash script tells the **bash shell** what it should say and do
- Then what is a bash shell?

Bash Scripting - Why a Bash Script?

- A film script tells the actors what they should say and do
- A bash script tells the **bash shell** what it should say and do
- Then what is a bash shell?

Bash Scripting - What is Bash shell?

- Hence a shell in the computing world wraps around the delicate interior of an Operating system protecting it from accidental damage.
- It takes input from you in the form of commands, processes it, and then gives an output
- Two main types of shell in linux
  - The Bourne ShellÕ with prompt $ (subtypes are bash and sh)
  - ÕThe C ShellÕ with prompt % (subtypes are csh and tcsh)

Bash Scripting - What is Bash shell?

- Hence a shell in the computing world wraps around the delicate interior of an Operating system protecting it from accidental damage.
- It takes input from you in the form of commands, processes it, and then gives an output
- Two main types of shell in linux
  - The Bourne ShellÕ with prompt $ (subtypes are bash and sh)
  - ÕThe C ShellÕ with prompt % (subtypes are csh and tcsh)
- So can you check which of the shells you are currently using?

Bash Scripting - How do we run them?

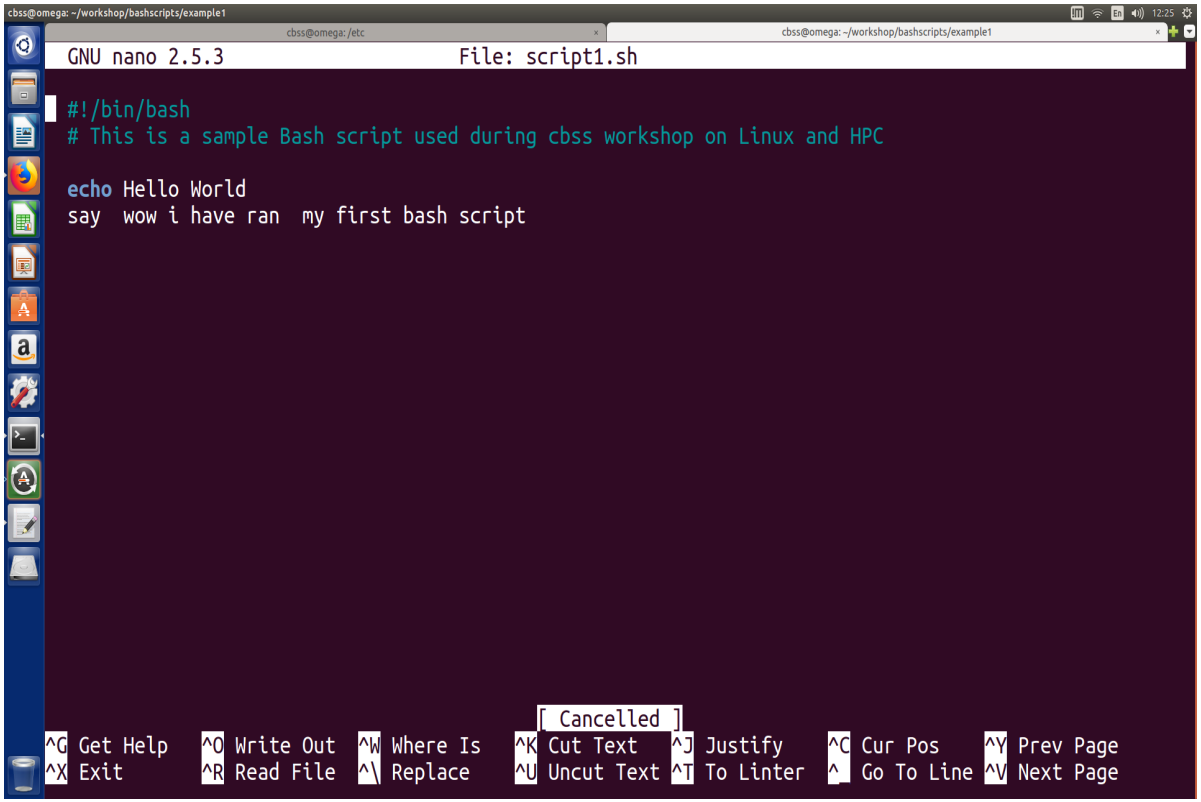
- Before we can execute a script it must have the execute permission set (for safety reasons this permission is generally not set by default)
- In this workshop, we'll be using the example bash scripts (with .sh extension) in /home/cbss/workshop/bashscripts/example1/
- Can you check if the file, script1.sh has an execute permission? If it doesn't have any, can you make it have one?

Bash Scripting - How do we run them?

- Before we can execute a script it must have the execute permission set (for safety reasons this permission is generally not set by default)
- In this workshop, we'll be using the example bash scripts (with .sh extension) in /home/cbss/workshop/bashscripts/example1/
- Can you check if the file, script1.sh has an execute permission? If it doesn't have any, can you make it have one?
  $ **ls -l script1.sh**
  $ **chmod +x script1.sh**
  $ **ls -l script1.sh**
- Finally, to run, type $ **./script1.sh**
- Were you successful

Bash Scripting - How do we run them?

- Please open the bash script, script1.sh with a text editor, we will fully discuss it's content (you can compare with a film script)



Bash Scripting - How do we run them?

- You noticed any difference between scripts and ordinary linux commands like **ls, pwd, cd etc**?

Bash Scripting - How do we run them?

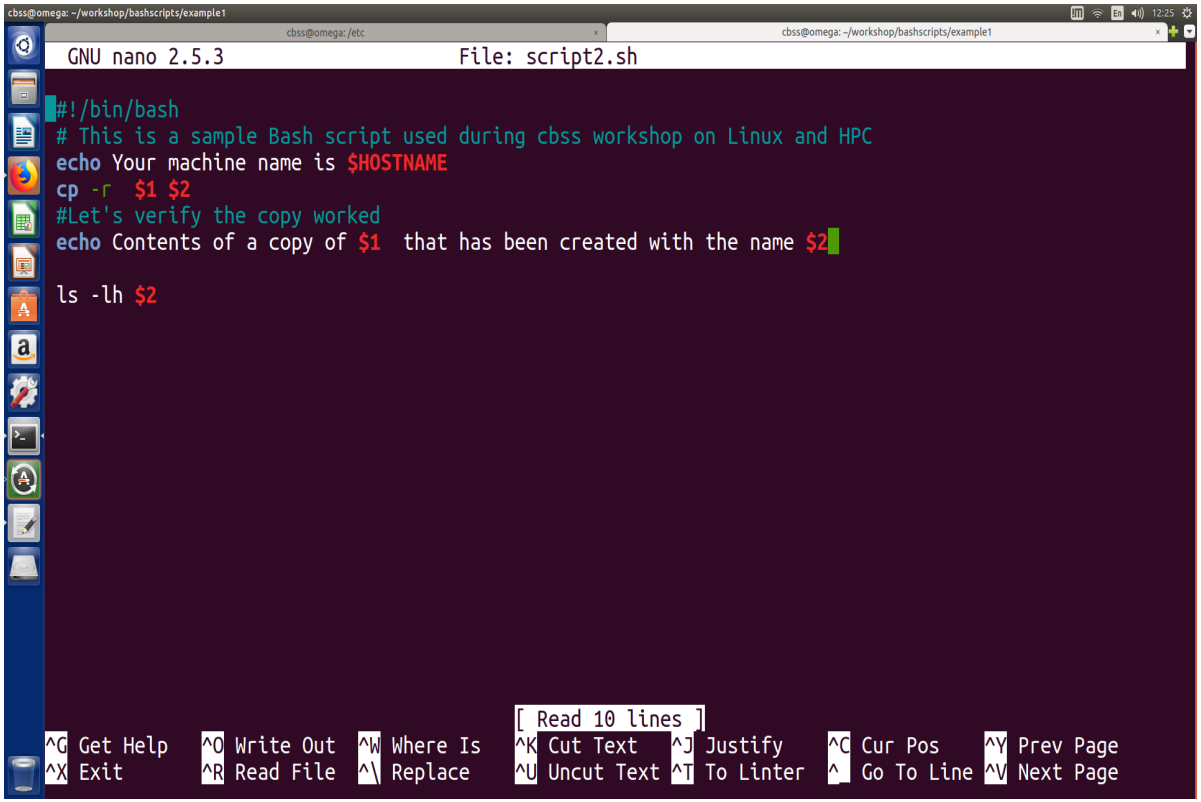
- You noticed any difference between scripts and ordinary linux commands like **ls, pwd, cd etc**?
- Yes, why can't I like run the script by typing only $ **script1.sh**?, like i used to do for other linux commands, e.g. $ **cd**
- These other Linux commands, **ls, pwd, cd etc** are all in a location searched by the environment variable called **PATH** (You recall the command **env**)
- Can I add the location/path of script1.sh to the paths searched by **PATH**? If yes, how

Bash Scripting - How do we run them?

- You noticed any difference between scripts and ordinary linux commands like **ls, pwd, cd etc**?
- Yes, why can't I like run the script by typing only $ **script1.sh**?, like i used to do for other linux commands, e.g. $ **cd**
- These other Linux commands, **ls, pwd, cd etc** are all in a location searched by the environment variable called **PATH** (You recall the command **env**)
- Can I add the location/path of script1.sh to the paths searched by **PATH**? If yes, how
- Yes you can. Take Home Assignment

Bash Scripting - Variables

- Two actions on variables - SETTING a value and READING a value
- As we did for script1.sh, do the same for script2.sh



- Repeat above while you try adding your own variable

Bash Scripting - Command Substitution

- Allows us to take the output of a command or program (what would normally be printed to the screen) and save it as the value of a variable
- As we did for script1.sh and script1.sh, do the same for script3.sh
- Were you successful ?

Bash Scripting - User Input

- If we would like to ask the user for input then we use a command called **read**. This command takes the input and will save it into a variable
- As we did for other scripts, do the same for script3.sh and scoreipt4.sh (many inputs)
- Were you successful ?

Bash Scripting - Arithmetic

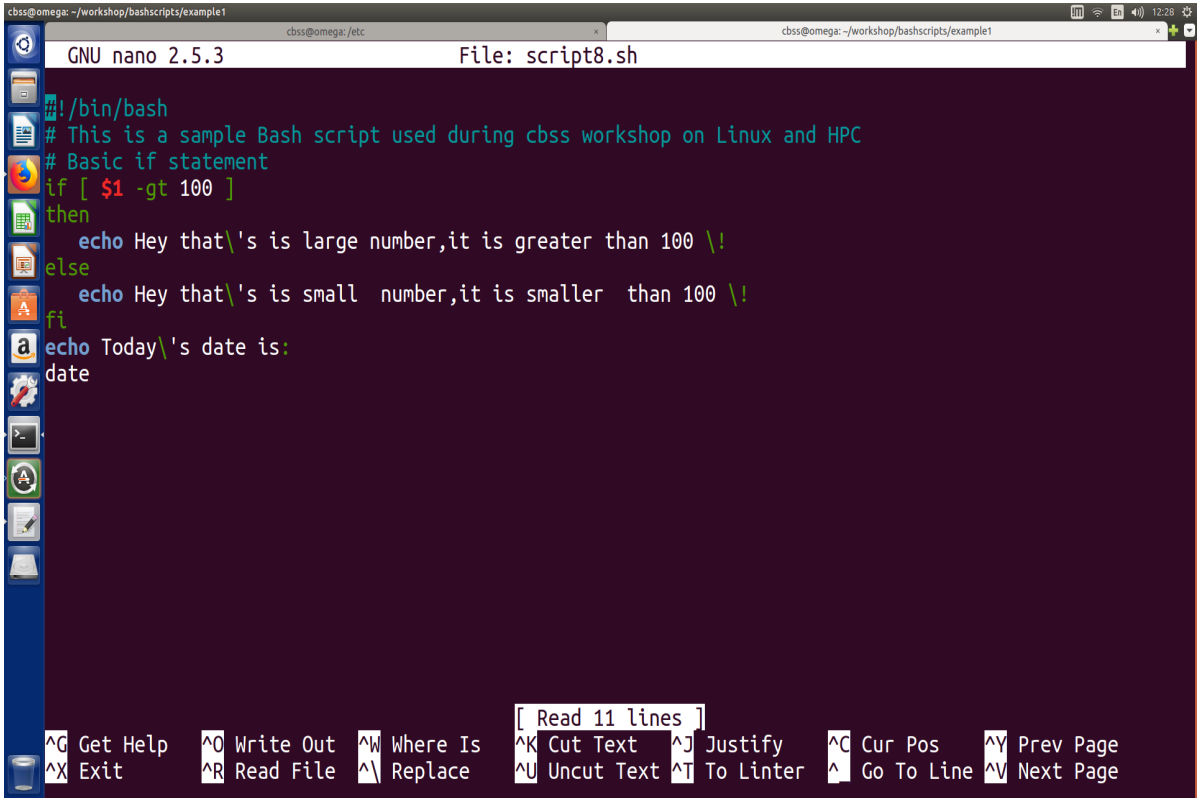
- let is a builtin function of Bash that allows us to do simple arithmetic. It follows the basic format:
  let ⟨ arithmetic expression ⟩
- Try script6.sh

Bash Scripting - if statements

- Allow us to decide whether or not to run a piece of code based upon conditions that we may set.
- If statements, combined with loops allow us to make **much more complex** scripts which may solve larger tasks.
- Study the content of script7.sh and try running it
- You noticed any similarities between the syntax of if statements in python and those in bash scripts?

Bash Scripting - if else statements

- Study the content of script8.sh and try running it



Bash Scripting - Boolean Operations

- Sometimes we only want to do something if multiple conditions are met.
- Other times we would like to perform the action if one of several condition is met. We can accommodate these with boolean operators and - && or - ——
- Study the content of script9.sh and try running it
- Were you successful? (gamma machine)

Bash Scripting - Loops

- Take Home Assignment - **while, for until** loops

Bash Scripting - Functions

- Functions in Bash Scripting are a great way to **REUSE** code

- Study the content of script10 .sh and try running it

Bash Scripting - Crontab For Job Automation and Scheduling

- Crontab stands for "cron table". It allows to use job scheduler known as cron to execute tasks.
- It is driven by a crontab file, a config file that indicates shell commands to run periodically for the specific schedule

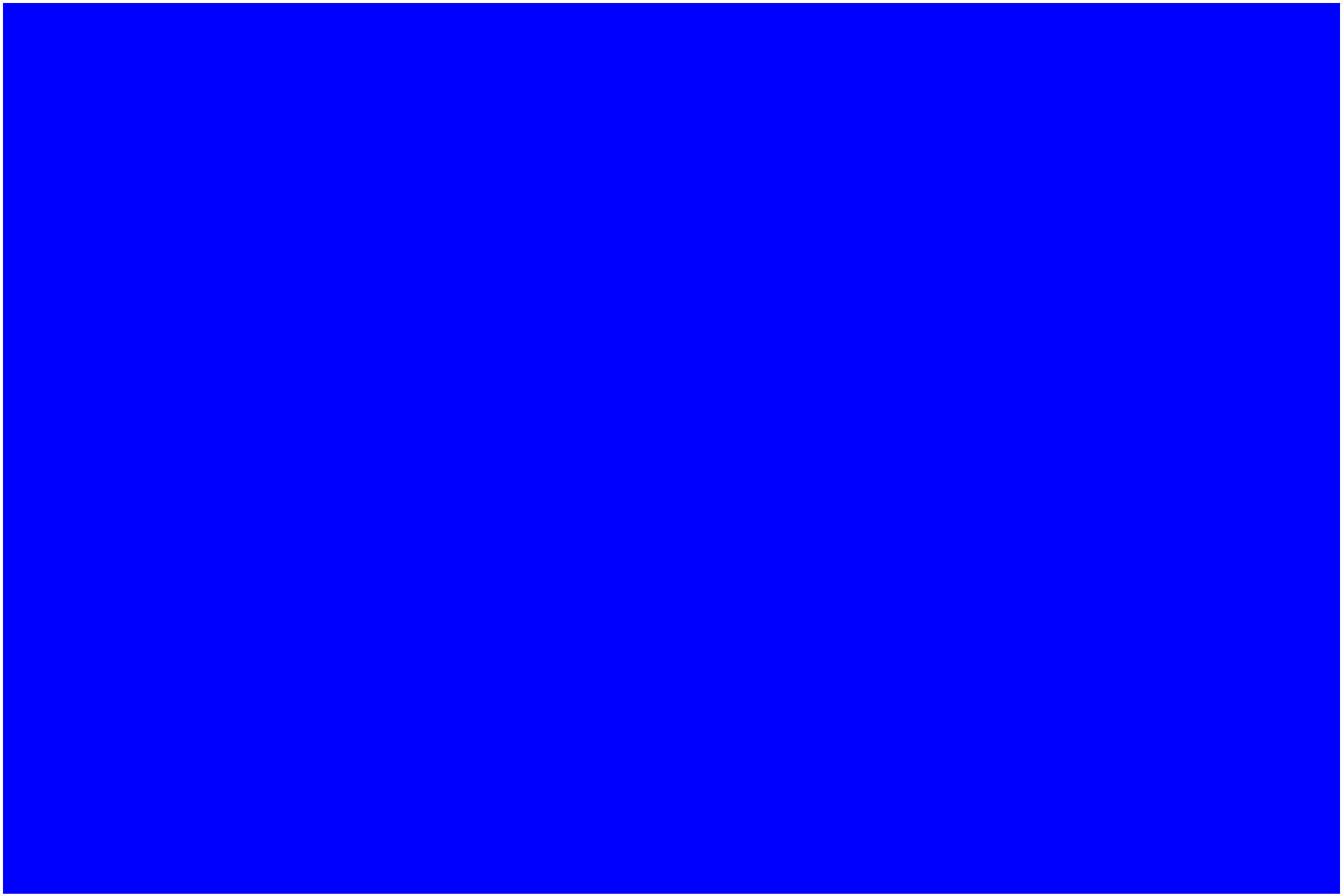
Bash Scripting - Reasons for using Cron jobs in Linux

- Helps OS to take a scheduled backup of log files or database.
- Delete old log files
- Archive and purge database tables
- Send out any notification email such as Newsletters, Password expiration email
- Regular clean-up of cached data
- An ideal option to automate Unix jobs.
- Automate system maintenance

Bash Scripting - Running a cron job

- Eg 1 is in script1.sh which performs backup of files. Edit crontab with $ **crontab -e**
- Eg 2 is in script12.sh - Take home assignment - but run during the workshop to produce a pdf output

```
# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR
sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
```



Introduction Python Programming Language

- CONCISE - Line of codes about a quarter of the length of a C program
- INTUITIVE -Easy guess on what you need to type to do something you want to get scripted
- FREE- No spending on licence
- HIGH-LEVEL - Lots of ready-made functions mainly from community, leaving you to concentrate on the problem you want to solve

Lets try out some scripts in "example_python_scripts"