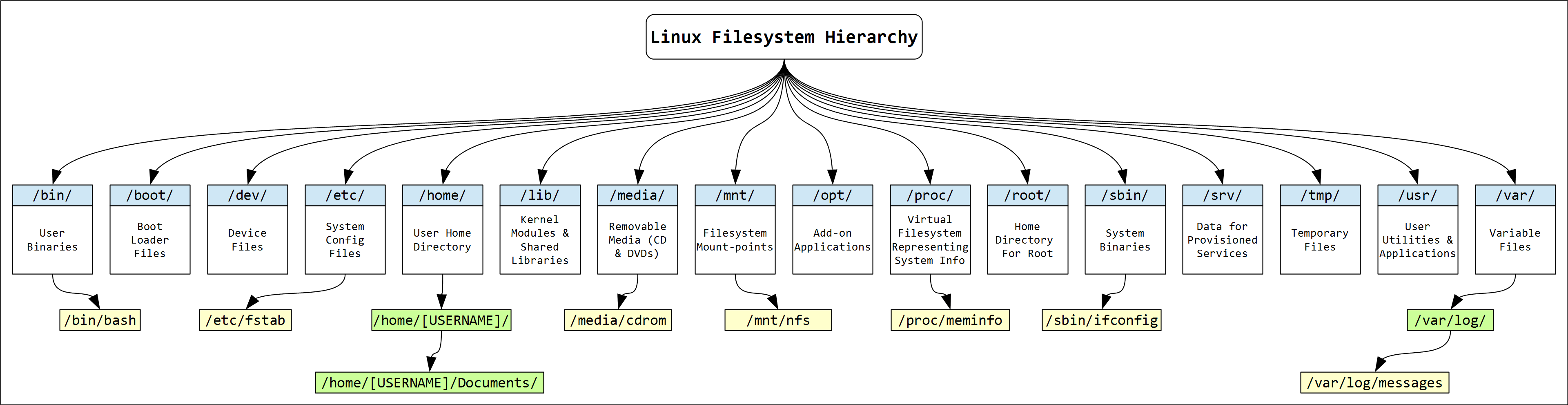
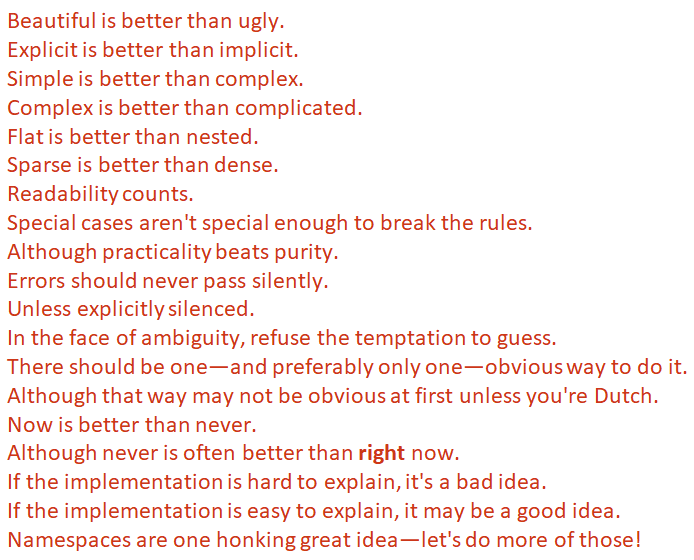
* **Linux** was first developed by Linus Torvalds in 1991.
* Today, the Linux operating system is used in operating systems for a wide range of devices. Linux is used in: desktops, servers, smartphones, routers, smart home devices and even smart cars.
* A number of developers have **modified** Linux.
  + Examples of these operating systems include: Android, WebOS (LG’s smart TV OS) and Raspbian.
  + It is a requirement that new operating systems built using Linux themselves be open source and, therefore, the previously mentioned operating systems are also open source.
* According to a survey of cloud servers in 2016, the most popular server Linux distribution, by a significant margin, is Ubuntu (Server) with 58% of the market share.
* Red Hat Enterprise Linux (RHEL) and CentOS were second and third with 21% and 13% of the market share, respectively.
* Windows Server made up just 2%.
* In this course we focus on CentOS.
* Ubuntu has a split between desktop and server users with a number of differences between the versions of the OS.
* RHEL requires a paid license, even for academic/educational users.
* CentOS has been owned by Red Hat since 2014 and CentOS itself is based off the source code of RHEL meaning many technical details are common across the two systems.
* We will be using CentOS 7 as the Linux distribution for this course.
* A **minimal installation** of CentOS 7 does not come with a graphical user interface (GUI)—all tasks are performed in the terminal.
* We will be using the KDE Plasma desktop environment for GUI functionality.





Most programming languages have six elements in common.

* Output
* Variables
* Input
* Maths
* Conditionals (If (something is true) then (do something))
* Iteration

Some of programs

1.

name = "Ben Martini"

print (name)

2.

name = input(What is your name? ')

3.

number1 = 2

number2 = 4

number3 = number2 + number4

print (number3)

4.

number1 = 2

number2 = 4

if (number1 > number2) : print ("Smaller")

5.

number1 = 2

number2 = 4

if (number1 > number2) :

print ("Answer: ")

print ("Smaller")

6.

x = 0

while (x < 5):

print('Hello World')

x = x + 1

print('Done')

7.

lengthInMetres = input('Length in metres: ')

lengthInMetres = float(lengthInMetres)

lengthInFeet = lengthInMetres \* 3.28084

lengthInFeet = round(lengthInFeet, 2)

widthInMetres = input('Width in metres: ')

widthInMetres = float(widthInMetres)

widthInFeet = widthInMetres \* 3.28084

widthInFeet = round(widthInFeet, 2)

areaInFeet = round(widthInFeet \* lengthInFeet, 2)

print('Length:\t', lengthInFeet, 'feet')

print('Width:\t', widthInFeet, 'feet')

print('Area:\t', areaInFeet, 'square feet')

8.

Pwd:- command to list your current working directory.

$ pwd/home/student

Ls :- command to list files in the current directory.

$ lsDesktop Documents Download Music Pictures Public Templates Videos

A :- flag to list all files in your current directory, including hidden files and folders.

$ ls –a

Help :- to print the help page

Man :- command with the argument ls to bring up the manual for the ls command.

$ man ls

Cd :- command is used to change your current working directory.

$ cd

Nano :- We can edit text files with the nano command. The command takes one optional argument - the path of the file toopen. If no file is specified, a blank file will be opened. nano can also create files - we will open/createmyfirstfile.txt:

$ nano myfirstfile.txt

Cat :- command se the cat command to concatenate (print) the contents of your file to the screen. It takesan unlimited number of optional arguments, each one a file that will be output. Print out the contents of your file.

$ cat myfirstfile.txt

Cp :- We can copy files with the cp command.

$ cp myfirstfile.txt mysecondfile.txt

Rm :- command to remove (delete) files.

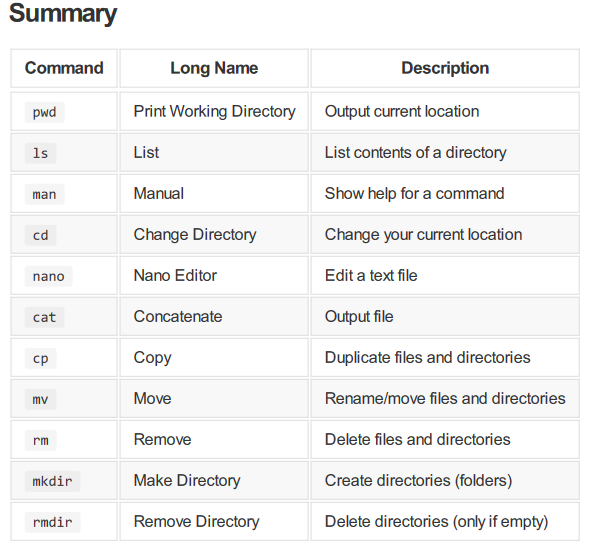
$ rm myfirstfile.txt

Mkdir :- command is used to create (make) directories.

$ mkdir myfirstdirectory

Rmdir :- To delete (remove) directories.

$ rmdir Directory With Spaces



Open the file, Write to the file, or read the data and Close the file

Opening: Four options

* r: Read file
* w: Write file
* a: Append file
* r+: Read and write file

1.

file = open('test.txt', 'w') **“Opens a "test.txt" file to write. If the file does not exist, it creates it first.”**

2.

file = open('test.txt', 'a') **“Opens an existing text file to append text to the end.”**

3.

file = open('test.txt', 'r+')  **“Allows both read and write.”**

4.

file = open('test.txt', 'w')

file.write('Hello\n')

file.write('World\n')

**“Use \n to start a new line.”**

Hello  
World

5.

animals =

['Dog','Tadpole','Axolotl','Turtle','Chicken']

file = open('test.txt', 'w')

file.writelines(animals)

DogTadpoleAxolotlTurtleChicken

6.

**Reading from a file**

**Method one: Read in the whole file.**

file = open('test.txt', 'r')

fileContents = file.read()

**Read in the contents of the file**

file = open('test.txt', 'r')

fileContents = file.read()

print(fileContents)

Line 1

Line 2

Line 3

7.

file = open('test.txt', 'r')

fileContents = file.readlines()

print(fileContents[1])

Line 2

**Read in all the lines into a list**

8.

animals =

['Dog','Tadpole','Axolotl','Turtle','Chicken']

file = open('test.csv', 'w')

for animal in animals:

file.write('"' + animal + '"' + ',')

file.close()

"Dog","Tadpole","Axolotl","Turtle","Chicken“

**Writes one row, each element in a new column.**

9.

gst = 0.1 **“set gst value”**

cost = input(“Enter cost :-”) **“enter cost”**

cost = float(cost)

finalprice = cost + (cost \* gst)

print(“Final price:”, finalPrice)

10.

Print(“Hello”)

If name == “Ben” :

print(“You work at UNISA”)

Elif name == ”scott” :

print(“You work as tutor”)

Else:

print(“I do not know where you work”)

print(“Goodbye”)

11.

text = "Gaye Martyni is a lecturer at UniSA."

if ("Martini" in text) :

print("The name is incorrect.")

else:

print("The name is correct.")

The name is correct.

* Here, we asks if it contains the word using “in”. “in” is great if we know exactly what we  
  want to match. “in” does fails if we are not matching an exact string.

12.

import re

text = "Kirsten Wahlstrom teaches INFS 5093."

m = re.search("INFS 5093", text)

if m :

print("A course is present.")

else:

print("A course is not present.")

A course is present.

* First, import the ”re” (Regular Expression) module.
* Now we can use re to search for some text.
* Here we saved the results of the search in “m”
* And here we ask what the result (“m”) is.

13.

import re

text = "Kirsten Wahlstrom teaches COMP 5093."

m = re.search("\w\w\w\w \d\d\d\d", text)

if m :

print("A course is present.")

else:

print("A course is not present.")

A course is present.

* \w means “word character”
* \d means “numeral”
* \w\w\w\w \d\d\d\d Means “four letters, a space, and four numerals”
* \* Means “0 or more”\w\* \d\* will match “0 or more letters, followed by a space, followed by 0 or more numerals.
* \w\w\w\w \*\d\d\d\d Will match INFS 5093 and INFS5093

14.

import re

text = "Kirsten Wahlstrom teaches INFS 5093."

m = re.search("\w{4} \*\d{4}", text)

if m :

print("A course is present.")

else:

print("A course is not present.")

A course is present.

* {n} Means “n or more times”
* \w{4} \d{4} will match “4 or more letters, followed by a space, followed by 4 or more numerals”.

15.

import re

text = "Kirsten Wahlstrom teaches INFS 5093."

m = re.search("\w{4} \*\d{4}", text)

if m :

print(m.group())

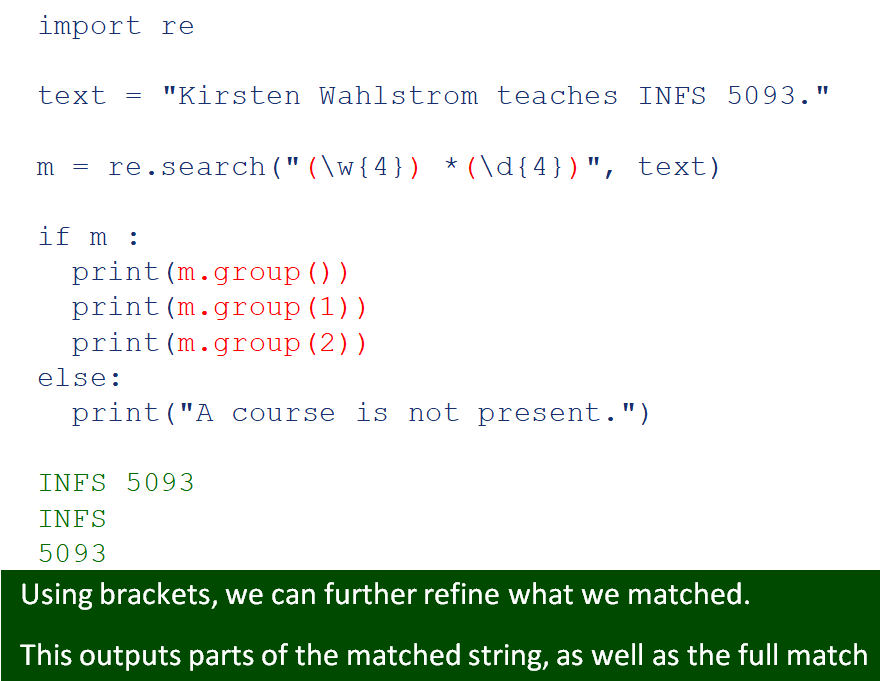
else:

print("A course is not present.")

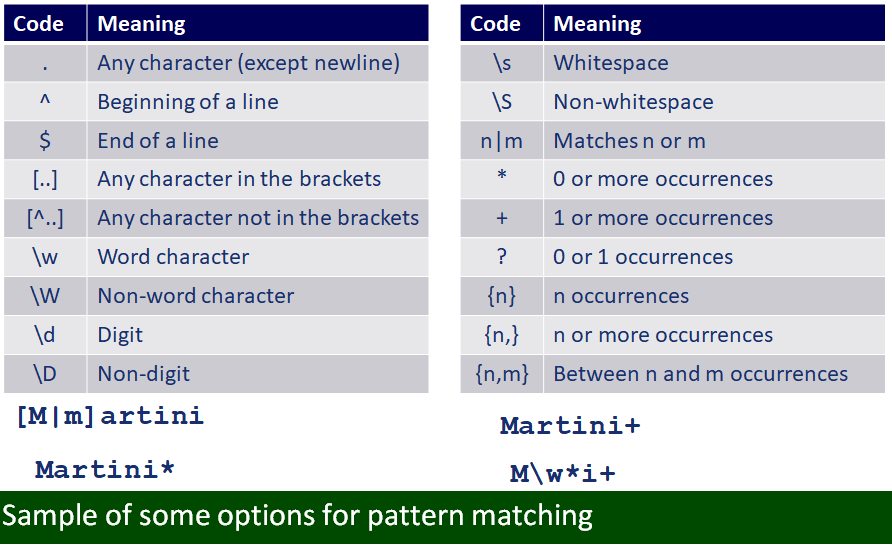
INFS 5093

* We can also find out what text was matched using the “group()” function.

16.



17.



**Python 2 vs Python 3**

* Python 2 was released in October 2000, and proved to be extremely popular
* 10 most popular programming languages.
* Python 3 was released in 8 years later in December 2008.
* at the time Python 2 was set to be completely retired in 2015.
* However, Python 2 and 3 are quite different, requiring significant chnages to programs to convert from one to the other. As a result the retirement date was postponed until 2020.
* The version of CentOS that we are currently using only comes with Python 2.

**Linux Binaries**

* All Linux commands are either binaries (compiled code) or scripts (text files).
  + For example, *yum* is executed as a Python script whilst *iptables* is a compiled binary.
* Linux programs are typically stored in one of the following locations:
  + **/bin/** – More commonly used commands.
  + **/sbin/** – Vital system binaries.
  + Local directories (e.g. /usr/bin/ and /usr/sbin/).