Question 1

a)

(I) **daemons** are background processes that run in the background. Most of these processes don't need human interaction; they start automatically when the operating system is started.

systemd is also a daemon

Without them, we wouldn't be able to do anything on GNU/Linux

Function 1

one of its functions is to schedule background jobs at regular intervals

for example, make daily backups and email weekly usage reports these are referred as

cron daemons

Function 2

it controls system networks, ssh, printers and more

(ii) The idea behind the Microkernel was only have the minimum essential elements such as **process and memory management**, and s**pecial interprocess communication IPC** mechanism most OS modules work in user mode and need to communicate with each other through **IPC** mechanism

(iii)

The salt string is less vulnerable to dictionary attacks because when the user sets a new password, the operating system generates a string of characters as the salt is added to the user's password.

When the password is checked the salt is again combined with the typed password before checking.

(iv)

because of additional data to append file to an existing file, you can't modify the file because the
next file occupies that space you could copy the original to make those changes however is time-consuming and leaves holes.

It would be for Linux log files where lines are continually added to the end of the file.

(v) 1 gives the user ability to automate repetitive tasks such as backups, updating software and more.

2 you can create an alias for some long (**CLI**) commands in Linux to reduce typing

these long commands.

b)

Race conditions are when an outcome depends on how fast each process executes and whichever wins the race to update the data that is been shared.
The outcome can differ because the winner can't be predicted, and there's no guarantee to be correct.
However, problems only arise if concurrent processes execute critical regions for shared resources.
A semaphore can lock a critical region so that only a single process can execute at a time.
For example, if a critical region is correctly identified, and concurrency avoided in this region, then correctness can be guaranteed, and race conditions will not occur.

(100 words)

c)

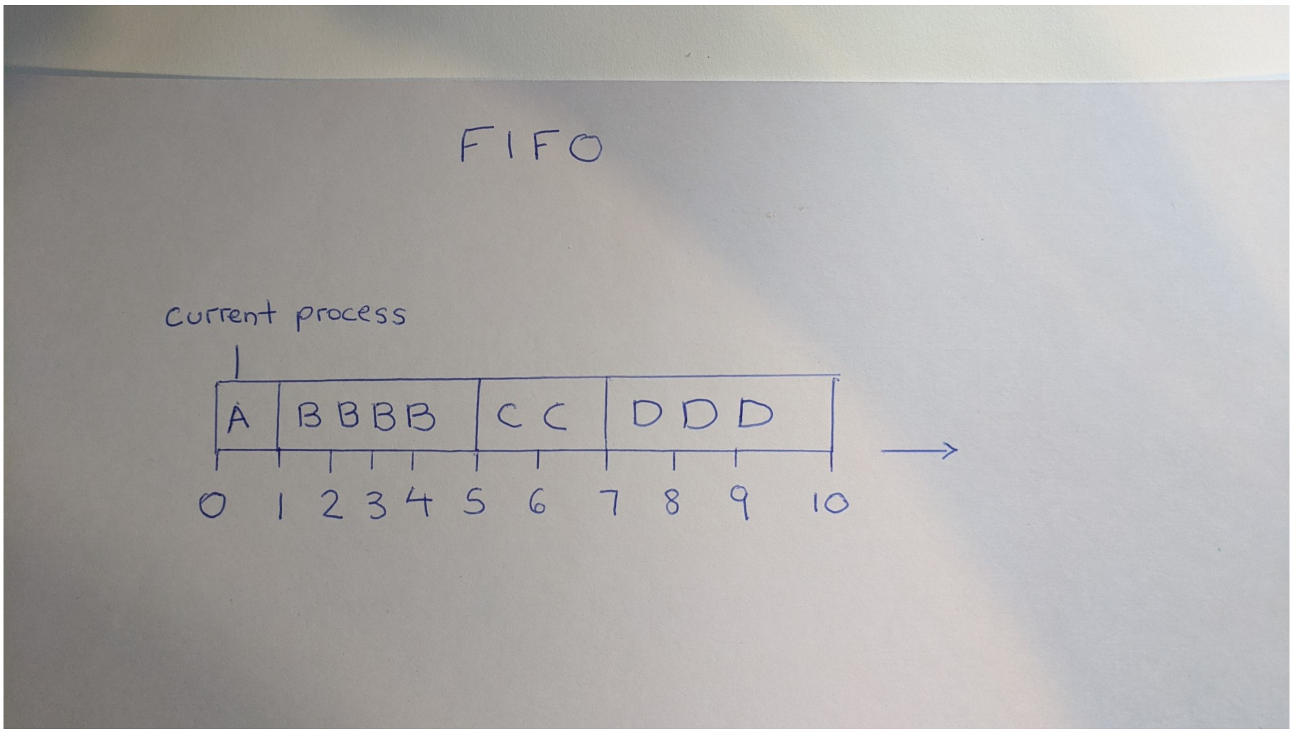
FIFO
shortcoming: a process could use the processor for a long period, preventing others processes to make progress.

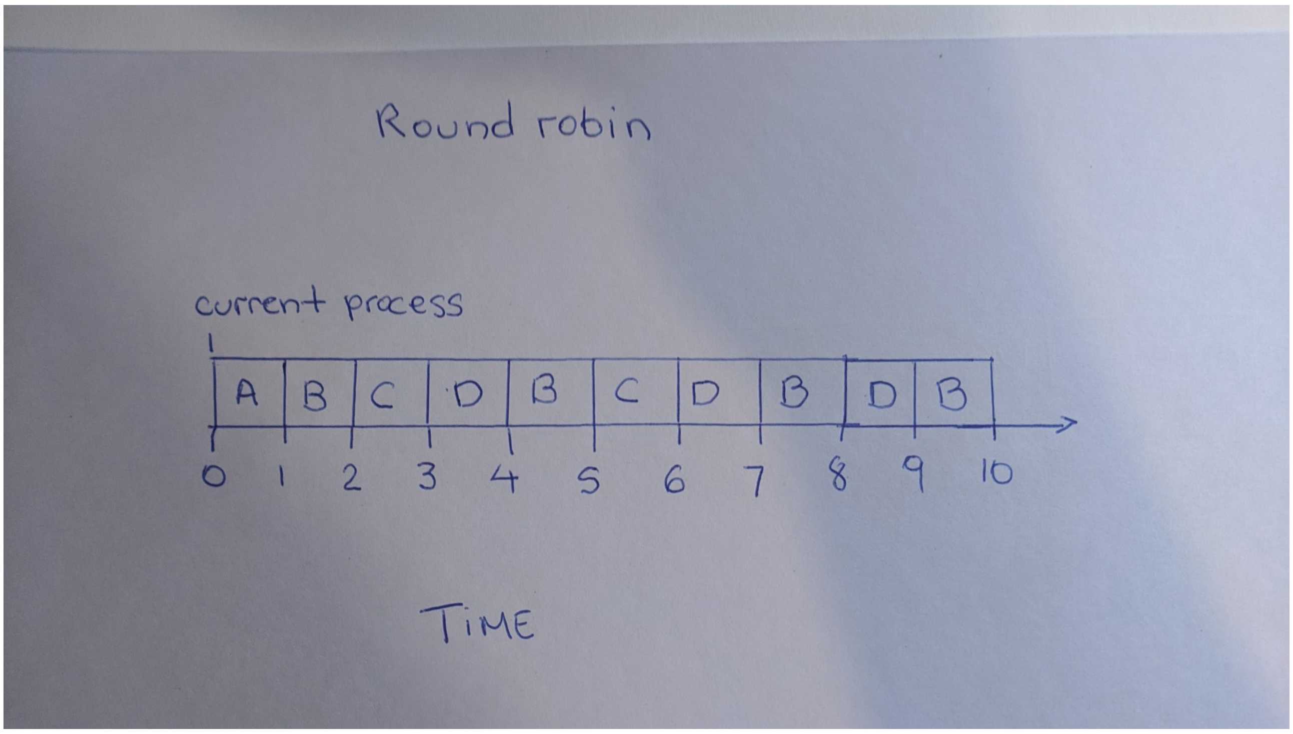
round-robin
shortcoming: some processors should have preference over the others for example

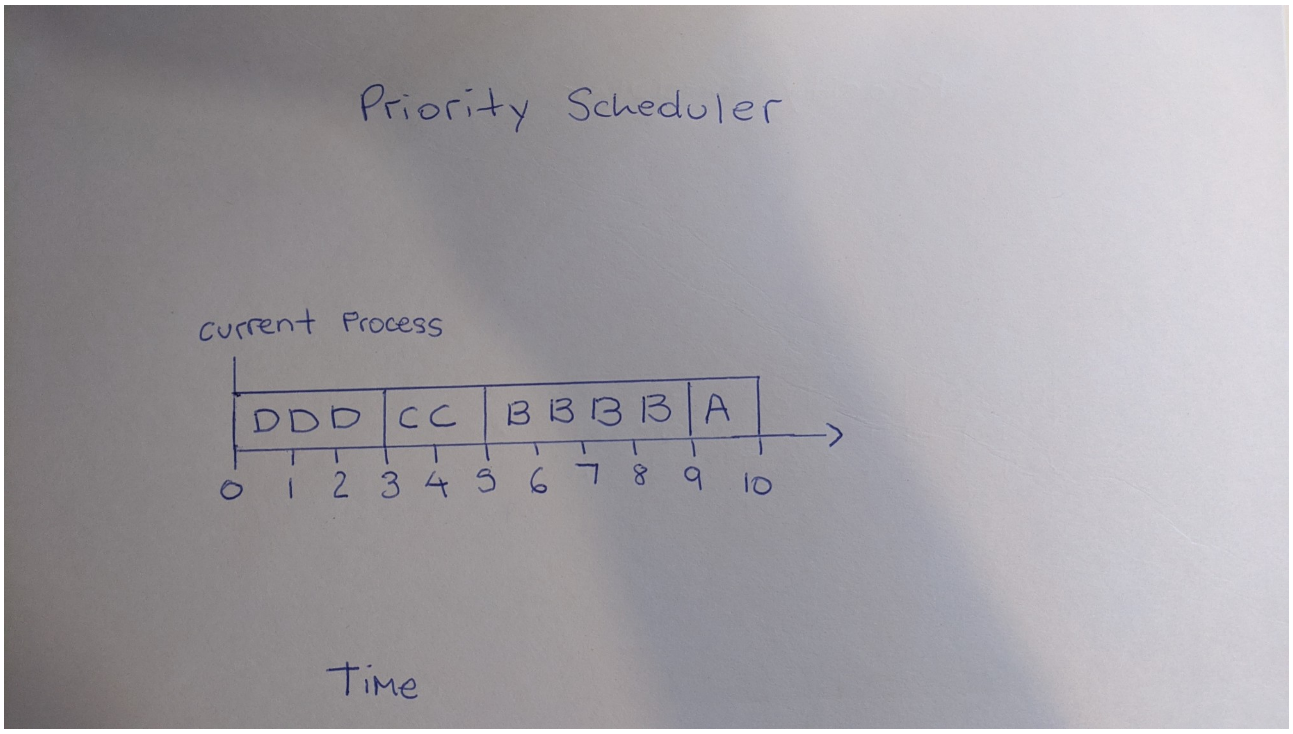
some processors are dedicated to I/O devices and data may be lost if they are delayed in the the ready queue for a long period.

priority
shortcoming: it can be unfair for example a low-priority processor may never get a turn

d)




Round robin
current process
A
B
C
D
B
C
D
B
D
B
0
1
2
3
4
5
6
7
8
9
10
TIME

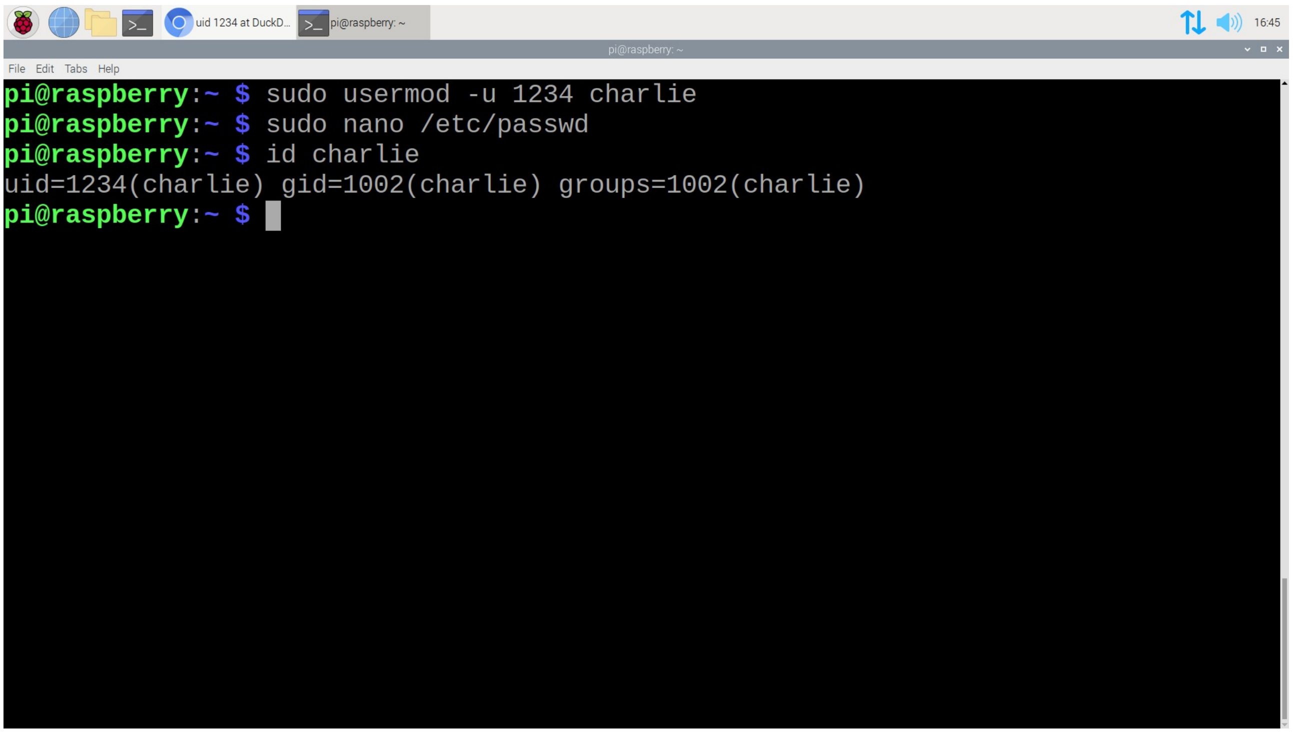


Question 2

a)

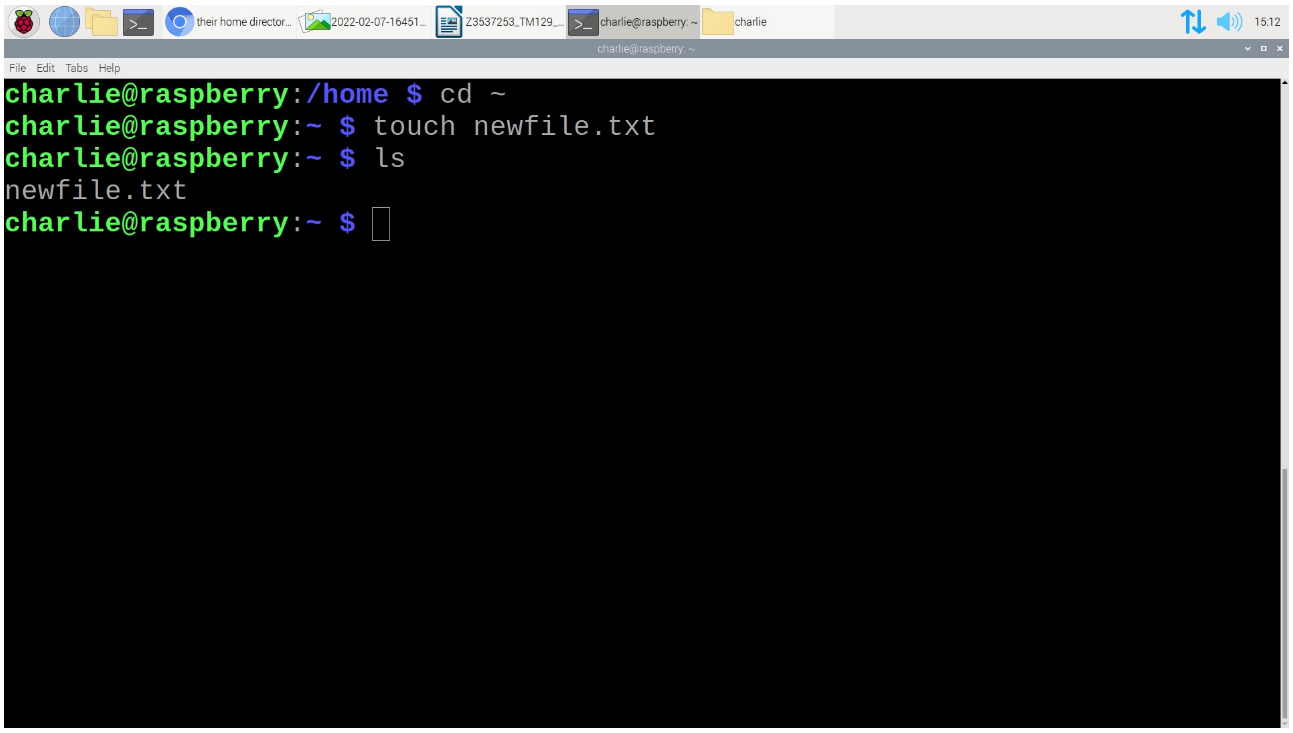
To achieve those tasks, I would look at the Linux forums, the Specific distro documentation for example, the Pi-Desktop website has good documentation about basic CLI commands, or I would use the man pages to get more information about a specific command.

b)



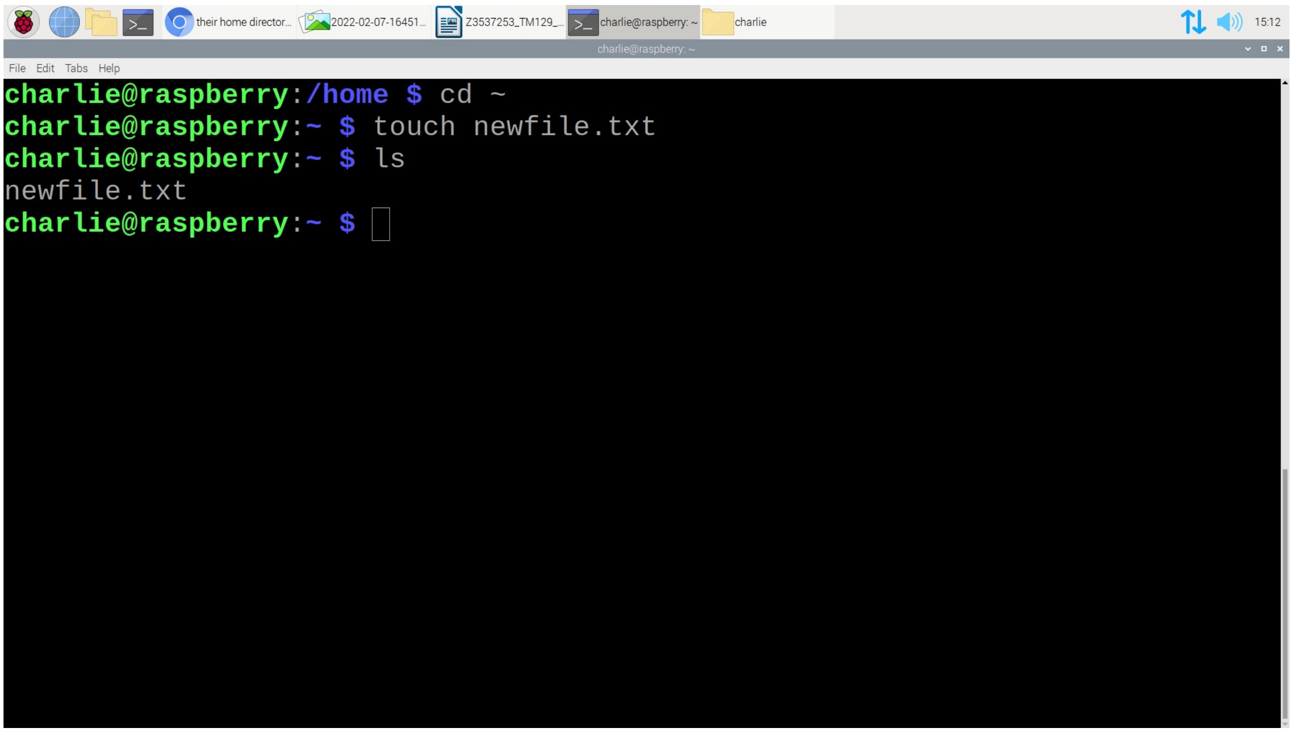
```
pi@raspberry:~ $ sudo usermod -u 1234 charlie
pi@raspberry:~ $ sudo nano /etc/passwd
pi@raspberry:~ $ id charlie
uid=1234(charlie) gid=1002(charlie) groups=1002(charlie)
pi@raspberry:~ $
```

c)



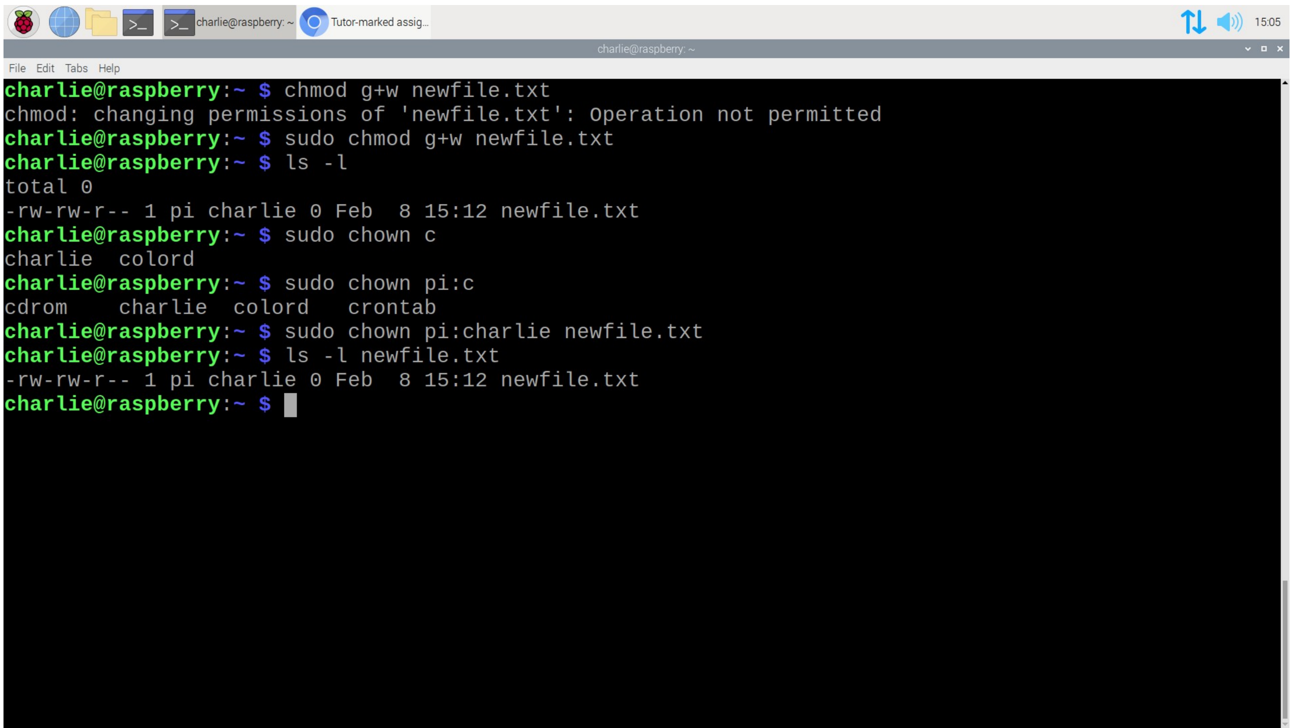
```
charlie@raspberry:/home $ cd ~
charlie@raspberry:~ $ touch newfile.txt
charlie@raspberry:~ $ ls
newfile.txt
charlie@raspberry:~ $
```

d)



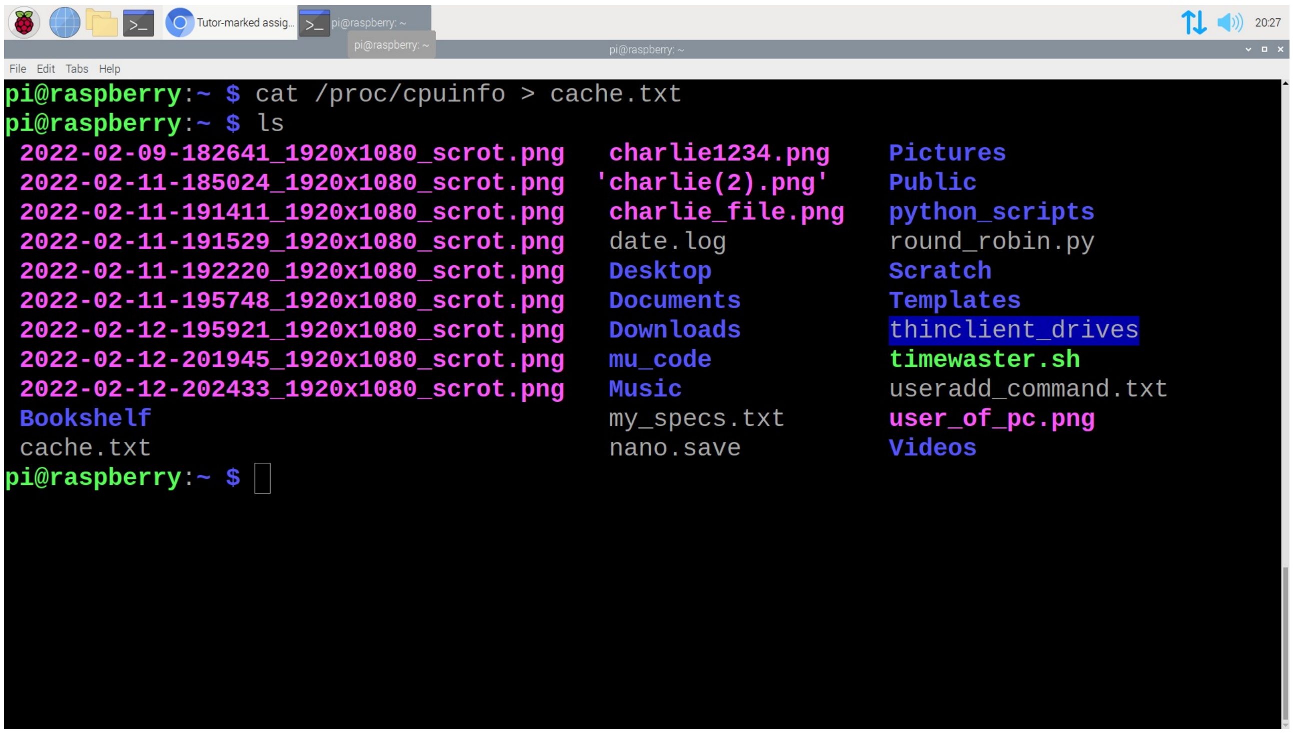
```
charlie@raspberry:/home $ cd ~
charlie@raspberry:~ $ touch newfile.txt
charlie@raspberry:~ $ ls
newfile.txt
charlie@raspberry:~ $
```

e)

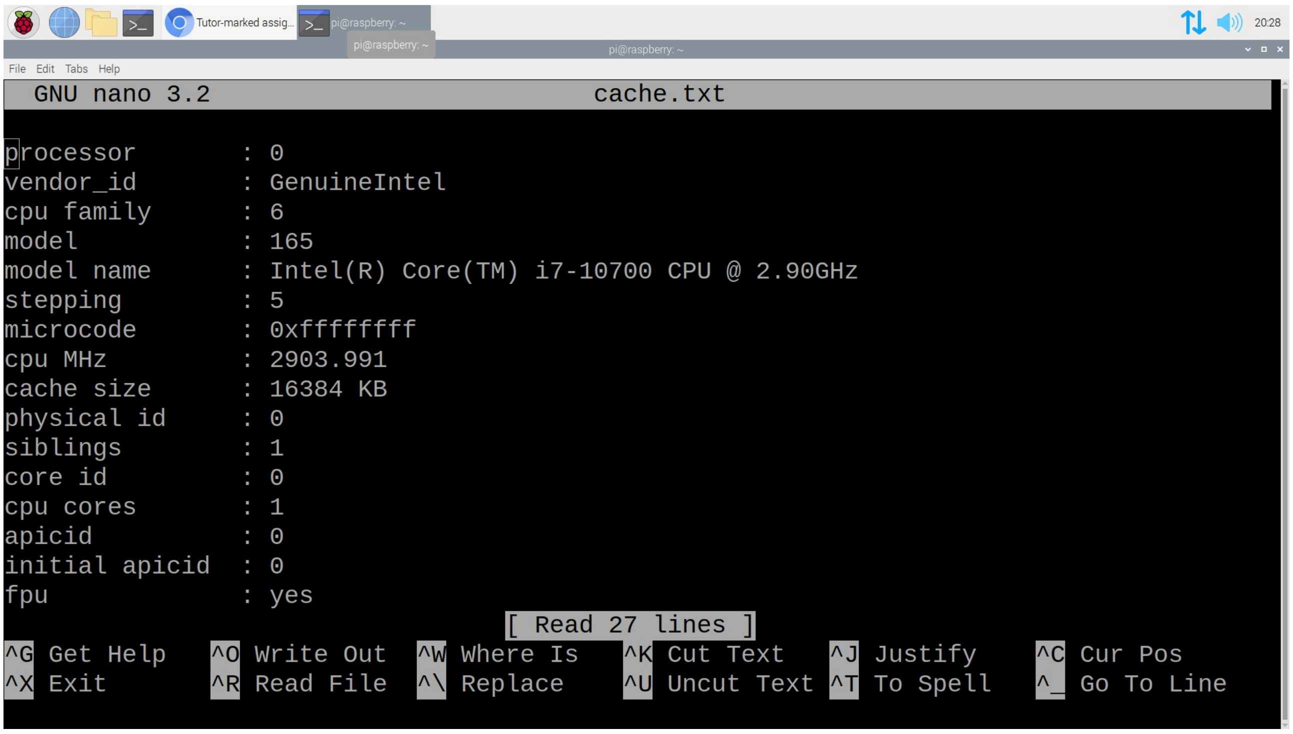


```
charlie@raspberry:~ $ chmod g+w newfile.txt
chmod: changing permissions of 'newfile.txt': Operation not permitted
charlie@raspberry:~ $ sudo chmod g+w newfile.txt
charlie@raspberry:~ $ ls -l
total 0
-rw-rw-r-- 1 pi charlie 0 Feb  8 15:12 newfile.txt
charlie@raspberry:~ $ sudo chown c
charlie  colord
charlie@raspberry:~ $ sudo chown pi:c
cdrom    charlie  colord   crontab
charlie@raspberry:~ $ sudo chown pi:charlie newfile.txt
charlie@raspberry:~ $ ls -l newfile.txt
-rw-rw-r-- 1 pi charlie 0 Feb  8 15:12 newfile.txt
charlie@raspberry:~ $
```

f)



```
pi@raspberry:~ $ cat /proc/cpuinfo > cache.txt
pi@raspberry:~ $ ls
 2022-02-09-182641_1920x1080_scrot.png   charlie1234.png     Pictures
 2022-02-11-185024_1920x1080_scrot.png  'charlie(2).png'     Public
 2022-02-11-191411_1920x1080_scrot.png   charlie_file.png    python_scripts
 2022-02-11-191529_1920x1080_scrot.png   date.log            round_robin.py
 2022-02-11-192220_1920x1080_scrot.png   Desktop             Scratch
 2022-02-11-195748_1920x1080_scrot.png   Documents           Templates
 2022-02-12-195921_1920x1080_scrot.png   Downloads           thinclient_drives
 2022-02-12-201945_1920x1080_scrot.png   mu_code             timewaster.sh
 2022-02-12-202433_1920x1080_scrot.png   Music               useradd_command.txt
 Bookshelf                               my_specs.txt        user_of_pc.png
 cache.txt                               nano.save           Videos
pi@raspberry:~ $
```



```
  GNU nano 3.2                                    cache.txt

processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 165
model name      : Intel(R) Core(TM) i7-10700 CPU @ 2.90GHz
stepping        : 5
microcode       : 0xffffffff
cpu MHz         : 2903.991
cache size      : 16384 KB
physical id     : 0
siblings        : 1
core id         : 0
cpu cores       : 1
apicid          : 0
initial apicid  : 0
fpu             : yes
                                  [ Read 27 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line
```

a)

Question 3

1 . Parrot OS

This is a **Debian** based distribution that comes with three options **Home edition** for Normal users, **Security edition** for cybersecurity experts and **IoT and cloud appliances edition** for IoT devices.

Parrot OS aims to make the life of its users more secure.

It has a bunch of tools for **software engineering**, **cyber security** and **digital forensics**

Parrot OS has a wide choice of sandboxing and powering options the system is updated frequently. The distro itself was designed through containerisation technologies such as podman and doc to make it compatible with many devices such as IoT devices
and it's a very lightweight distro perfect for older hardware. It can even run on a web browser

Features

1 It provides operating system support for Tor, I2P

2 It comes with Anti Forensic tools, cyptsetup interfaces and support for encryption tools such as VeraCrypt, Truecrypt and LUKS.

2. Kali Linux

This is another Debian based distribution that focuses on forensic experts and pen-testers it has a nice **GUI** that is very easy to use I prefer the Xfce Desktop environment.

Features

1- it comes with cyber security tools such as Wireshark, kismet, NMAP and many more applications.

2 - It comes with Tor Browser as well and It can be used as a Live USB drive
and it can run on almost anything, the minimum system requirements are:
32 bit or 64-bit processor
512 MB of RAM
10GB of Disk space

3. Kodachi Linux
This is an **Ubuntu-**based distro that runs well even from a USB drive or SD card.
It comes with VPN services already installed and the Tor browser for online privacy.
Features
1 – The distro comes with DNSCrypt for masking TCP/IP protocols so that all online activities are filtered through privacy filters.
2 – users can encrypt their file directories and emails with high-grade cryptography tools available within Kodachi's OS
My preferred distribution
I would install **Parrot OS** because is more beginner-friendly and the **GUI** is easier to use, Security updates are more frequently than the other distros, and I have it installed on Virtual Box and it comes with installed text editors, office suite Tor and more.
(367 words)
References:

Parrot OS
Linux Shout (2021) ".10 Best Linux Distros for Privacy and Security for 2022"
Available at: https://www.how2shout.com/linux/10-best-linux-distros-for-privacy-and-security-for-2022/
(Accessed: 18 February 2022).
Kali Linux
Secured You (2022) ".10 Most Secure Linux Distros for Security, Anonymity, Privacy 2022"
Available at: **https://www.securedyou.com/best-most-secure-linux-distros-for-security-anonymity-privacy/**
(Accessed: 18 February 2022).

Kodachi OS
MUO (2022) ". The 9 Best Linux Distros for Privacy-Focused Users"

Available at: **https://www.makeuseof.com/best-linux-distros-for-privacy-security/**
(Accessed: 18 February 2022).

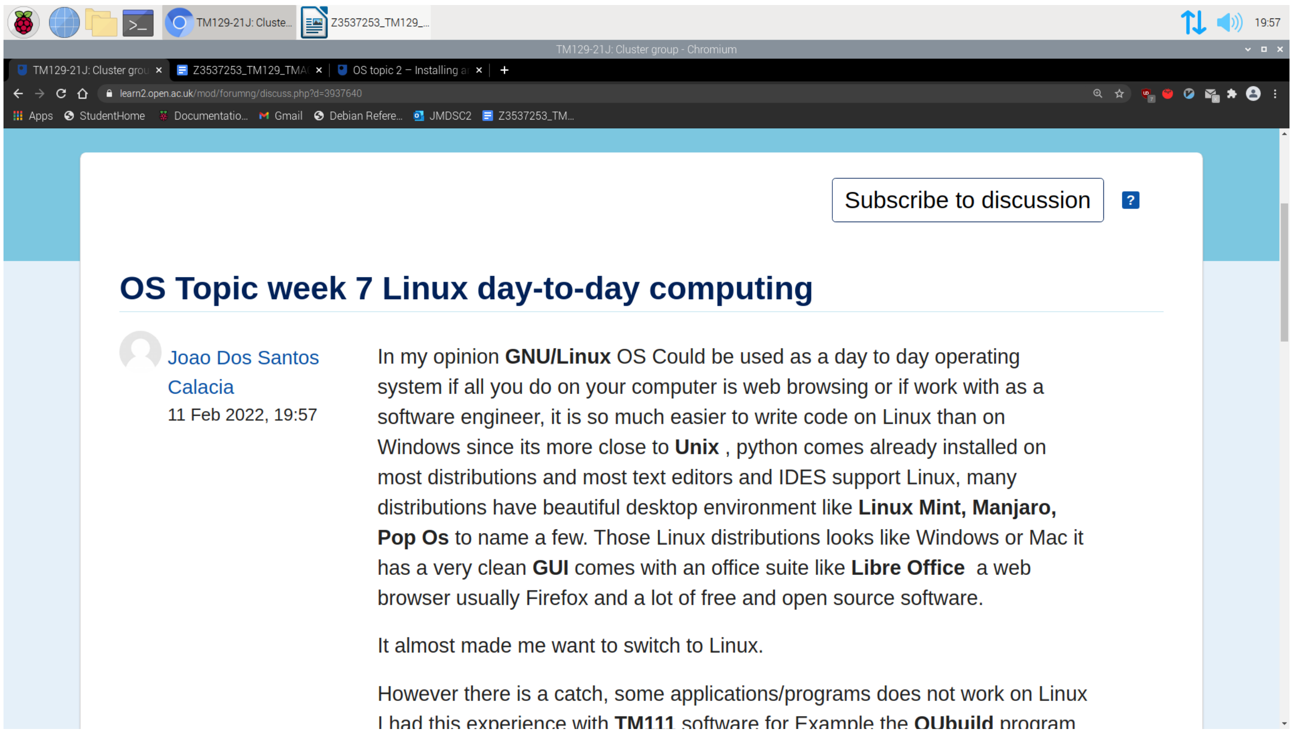
b)

One way to test **Linux** is to install it in a virtual machine also known as a hypervisor

There are many types of hypervisors software but the one that I use is **Virtual Box** because
It's free and relatively easy to use and you can install any **OS** on it.
It's a virtual computer running inside of your actual computer.
Advantage
you can run Linux inside Virtual Box as a virtual machine without the need to install it on your actual computer is very secure you can test Linux without commitments.
Disadvantage
If your computer has older hardware, then the performance will be
a problem because your VM will be sharing
(CPU and RAM) resources with your actual computer so both (VM) and your computer will be slow.
(129 words)
(496 words)

Question 4

a) Topic 2 Activity 10 (Linux day-to-day computing) this is my post on the Forum discussing using Linux as a day-to-day operating system and if the Pi Desktop could be a low-cost Desktop replacement the link to my discussion https://learn2.open.ac.uk/mod/forumng/discuss.php?d=3937640


TM129-21J: Cluster group - Chromium
learn2.open.ac.uk/mod/forumng/discuss.php?d=3937640
Subscribe to discussion

OS Topic week 7 Linux day-to-day computing

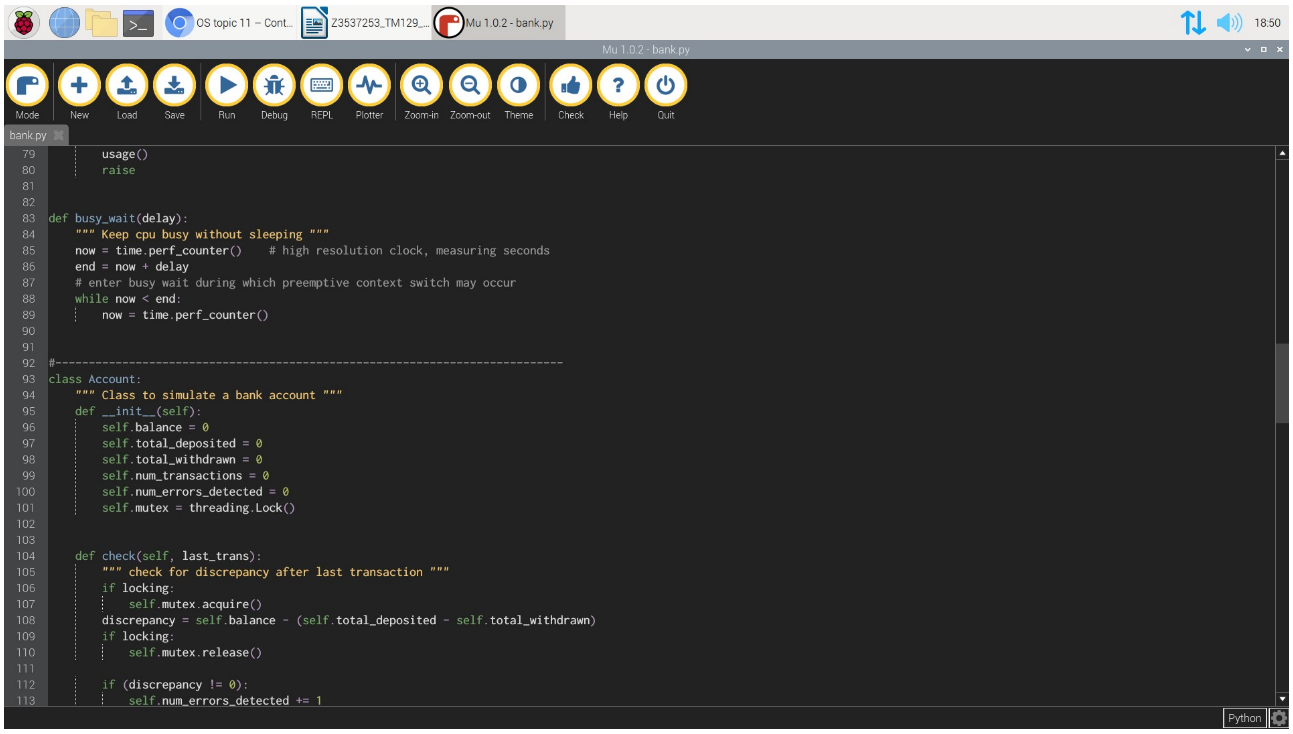
Joao Dos Santos Calacia
11 Feb 2022, 19:57

In my opinion **GNU/Linux** OS Could be used as a day to day operating system if all you do on your computer is web browsing or if work with as a software engineer, it is so much easier to write code on Linux than on Windows since its more close to **Unix** , python comes already installed on most distributions and most text editors and IDES support Linux, many distributions have beautiful desktop environment like **Linux Mint, Manjaro, Pop Os** to name a few. Those Linux distributions looks like Windows or Mac it has a very clean **GUI** comes with an office suite like **Libre Office** a web browser usually Firefox and a lot of free and open source software.

It almost made me want to switch to Linux.

However there is a catch, some applications/programs does not work on Linux I had this experience with **TM111** software for Example the **OUbuild** program

Topic 11 activity 10 on this activity I am exploring the python script (banking script) on the GUI text editor (MU)



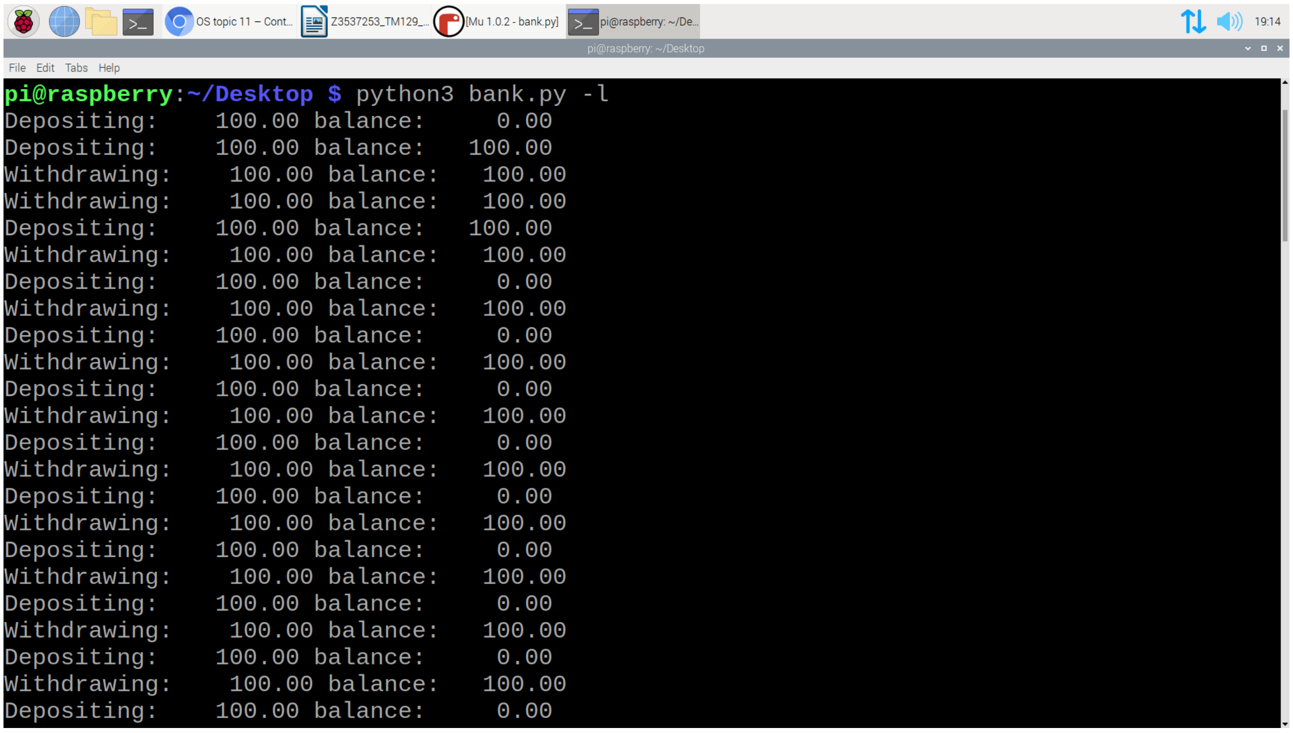
```
        usage()
        raise


def busy_wait(delay):
    """ Keep cpu busy without sleeping """
    now = time.perf_counter()    # high resolution clock, measuring seconds
    end = now + delay
    # enter busy wait during which preemptive context switch may occur
    while now < end:
        now = time.perf_counter()


#----------------------------------------------------------------------------
class Account:
    """ Class to simulate a bank account """
    def __init__(self):
        self.balance = 0
        self.total_deposited = 0
        self.total_withdrawn = 0
        self.num_transactions = 0
        self.num_errors_detected = 0
        self.mutex = threading.Lock()


    def check(self, last_trans):
        """ check for discrepancy after last transaction """
        if locking:
            self.mutex.acquire()
        discrepancy = self.balance - (self.total_deposited - self.total_withdrawn)
        if locking:
            self.mutex.release()

        if (discrepancy != 0):
            self.num_errors_detected += 1
```



```
pi@raspberry:~/Desktop $ python3 bank.py -l
Depositing:    100.00 balance:     0.00
Depositing:    100.00 balance:   100.00
Withdrawing:    100.00 balance:   100.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:   100.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
Withdrawing:    100.00 balance:   100.00
Depositing:    100.00 balance:     0.00
```

The image above is the script running on the terminal

b)

Activity 10 Linux day-to-day computing

This activity was interesting because I've got to play around with Linux and virtualization at the same time.

I have installed a bunch of Linux distributions on **Virtual Box,** and I made a challenge to just use Linux as my main computer, If I wanted to get software or update my system, I used the command line to do those tasks

I have even learned some CLI commands (**sudo apt update && sudo apt upgrade**) to update the system and **(sudo apt install <package-name>)** to install software on Debian Based distros and to be honest I could use Linux as my main desktop for **TM129,** unfortunately, some of the Open University software does not run on Linux which is the case of **OUbuild** a TM111 scratch version,

However, on activities 14, 16, 17 and 19 writing a bash script to make a backup of files was difficult for me I have never tried to automate any task by scripting I am not good at writing code as well. I struggled to make the backup script works I was always getting an error message "File or directory not found"

I am still learning some of the concepts of programming, I still don't understand it very well, even though I should by now.

(235 words)