**6G5Z1105**

**Computer Networks and Operating Systems**

Coursework - Operating Systems portfolio

**PART A: Report on Operating Systems**

Overall Description of the Kernel

The latest Windows kernel is NT version 10 (Thurrott, 2014) and the latest Linux kernel is version 5 (Linus, 2019). One big difference between these two is that Windows has a microkernel architecture whereas Linux uses monolithic kernel. The supported processor architectures for Windows version 10 are: IA-32 and x86-64 (Microsoft, 2017). Intel Architecture 32-bit (IA-32) is a 32-bit version of the x86 architecture created by Intel and AMD64/Intel64 (x86-64) is a 64-bit version of the x86 architecture. On the other hand, Linux version 5 supports a much wider variety of architectures such as: alpha, blackfin, arm, arm64, aavr, m68k, frv, hexagon, parisc, h8300, s390, ia64, x86, m32r, microblaze, mips, mn10300, openrisc, powerpc, sparc, arc, score, unicore32 and many more (Wikipedia, 2019). The latest mobile OS that uses a Windows NT kernel is ‘Windows 10 Mobile’ and for Linux it is ‘Android 10’ which uses versions 4.4, 4.9 or 4.14 of the Linux kernel (Wikipedia, 2020).

Detailed Comparison of Windows and Linux Kernels

Unlike Windows, Linux uses a variety of different file system types such as: EXT3, EXT4, BTRFS and XFS. Just to name a few. However, Windows only uses FAT and NTFS (Panwar, 2012). Linux and Windows both organise files on the disk into a hierarchy of directories, these hierarchies are called file systems. On Windows every file system is allocated a drive letter such as ‘D:’. On Linux each file system is allocated a device such as ‘/dev/sda’, this is represented as a file because on Linux “Everything is a File”, lots of different things such as all hardware devices are also represented as files (Hoffman, 2016). Every file system on Windows links to a file allocation table, this table holds information on which disk blocks contain the top directory, Linux doesn’t use a file allocation table although, it does use something similar; a superblock. Usually there are multiple copies of this superblock saved onto the disk, this is in case there is a partial disk corruption or something similar (McFarlane, 2004).

Windows uses a data structure called a Tree for managing its memory, each node of the Tree is called a Virtual Address Descriptor, the VAD in each node is set as either free, committed or reserved. As opposed to using a tree, Linux uses a Linked List for this (Kath 1993). One difference in memory management is that Linux uses Demand Paging meaning that pages are only brought into memory if the executing process needs them Windows, on the other hand, uses Cluster Demand Paging which works in a similar way however instead of pages being brought into memory one at a time, multiple pages are brought in at a time. In terms of page replacement algorithms Linux uses the Last Recently Used (LRU) algorithm whereas Windows uses the First in First Out (FIFO) algorithm. LRU works in a way that if a page is not used for a long period of time it is replaced and FIFO replaces the oldest page, but this means it is susceptible to Bélády's anomaly (Stallings, 2012).

All hardware requires a specific driver for it to be useable. All windows driver support is manufacturer specific but, in Linux driver support comes from the Kernel, the support comes in the form of modules that can be removed, added or changed very easily. This essentially means that Linux is chipset specific as opposed to Windows being manufacturer specific, therefore, Linux supports more hardware from the get-go as it only needs different drivers for each chipset whereas Windows needs different drivers for each individual piece of hardware (Locutus 2009).

Linux is open source and has a huge community around this, this means every user has access to the source code meaning that vulnerabilities are more likely to be detected earlier on. Also, if there is a security threat anyone can commit a fix for it but in Windows you are not able to change the source code and must wait until Microsoft fixes the security threat or vulnerability (Usually comes in the form of automatic updates). However, in terms of actual Linux users only about 2% of people use it in comparison to Windows enormous 88% this means hackers are incredibly more likely to target Windows based systems (Wikipedia, 2020 [For Statistics]). Another big reason Linux is far more secure than Windows is that it has a modular kernel whereas Windows does not this means it all parts of the Windows system are vulnerable when under an attack however, because of Linux modular design it is able to contain a threat to one part of the system. When using windows, the user has full admin privileges but in Linux user accounts to not normally have root access meaning that it is harder for malware to make drastic changes to the system. Linux also uses iptables to defend against network security risks, iptables are essentially a firewall utility that comes built in, it uses policy chains to allow or deny incoming traffic, there is a bit set of rules contained within the iptables and when a connection is establishing rules are located within the list to decide whether to allow or block the connection (Nexols, 2016).

In terms of reliability Linux is extremely reliable as it has brilliant system security, good process management and high uptime (for servers), every major internet company uses Linux for there servers. Windows is generally considered less reliable than Linux due to its many security flaws and system instability, Windows tends to slow down quite drastically after long term use or installing lots of different bits of software.

**PART B: BASH Programming**

Description

I decided to use one of the JD networking labs for these exercises, I chose 4 PCS in a row to work with. The flavour of Linux used was Debian which comes already installed on the JD computers. My script uses the ip: 149.170.10.$i loops through the node numbers and appends each one to the end of the ip. For example if I were using the 4 machines with ips: 149.170.10.19, 149.170.10.18, 149.170.10.17, 149.170.10.16, I would loop through numbers 19 to 16 (‘for i in {19..16}’)and append this to the ip: 149.170.10.$i.

Exercise 1: Ping Three Network Nodes

*Bash Script Code (ex1\_script.sh):*

#!/bin/bash

for i in {19..16} #loops through 'nodes'

do

ping -c 1 149.170.10.$i #pings node

if [ $? -eq 0 ]; then #checks return value of ping command if code '0' then success else no reply or errors.

echo "node 149.170.10.$i is on" >> ./ping\_test.txt #writes status to ping\_test.txt

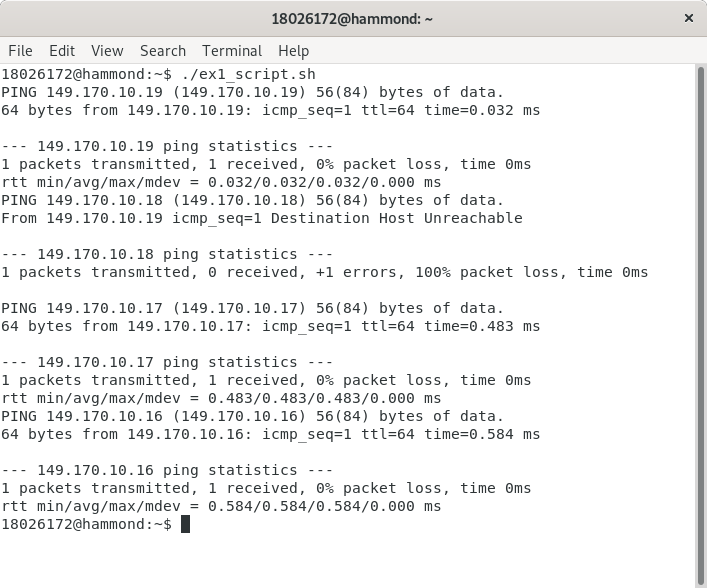
else

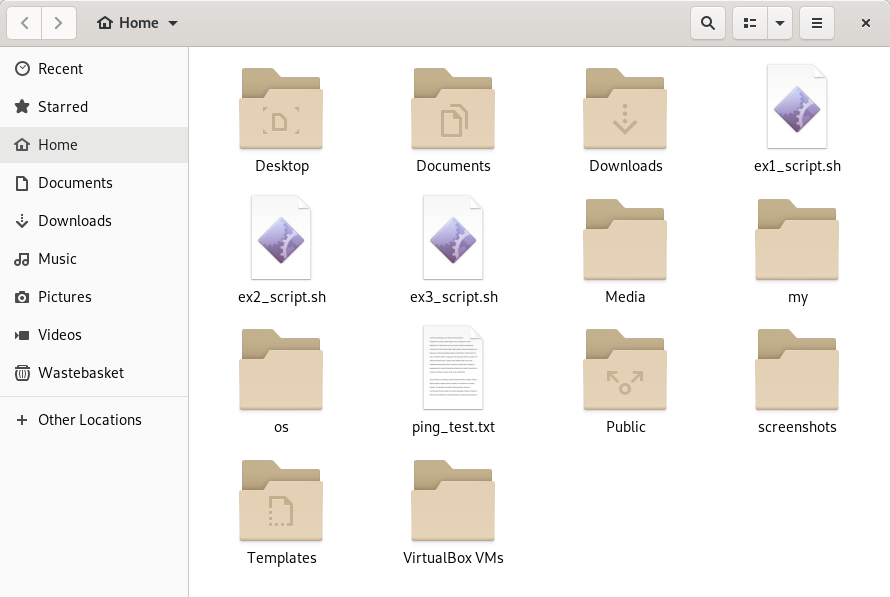
echo "node 149.170.10.$i is off" >> ./ping\_test.txt

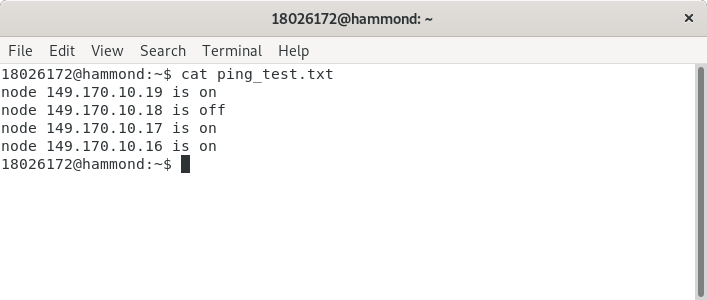
fi #end if

done #end loop

*Screenshots*







Exercise 2: Extracting Local System Information

*Bash Script Code (ex2\_script.sh):*

#!/bin/bash

mkdir OS\_details #create directory OS\_details

hostname >> OS\_details/info.txt #writes host name to text file

uname -r >> OS\_details/info.txt #writes kernel version and architecture to text file

lscpu | grep 'CPU family' | awk '{print $3}' >> OS\_details/info.txt #writes CPU family to text file

lscpu | grep 'CPU max MHz' | awk '{print $4}' >> OS\_details/info.txt #writes CPU MHz to text file

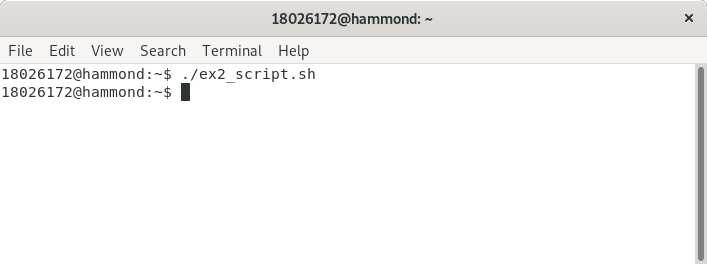
lscpu | grep 'Core(s) per socket' | awk '{print $4}' >> OS\_details/info.txt #writes number of cores to text file

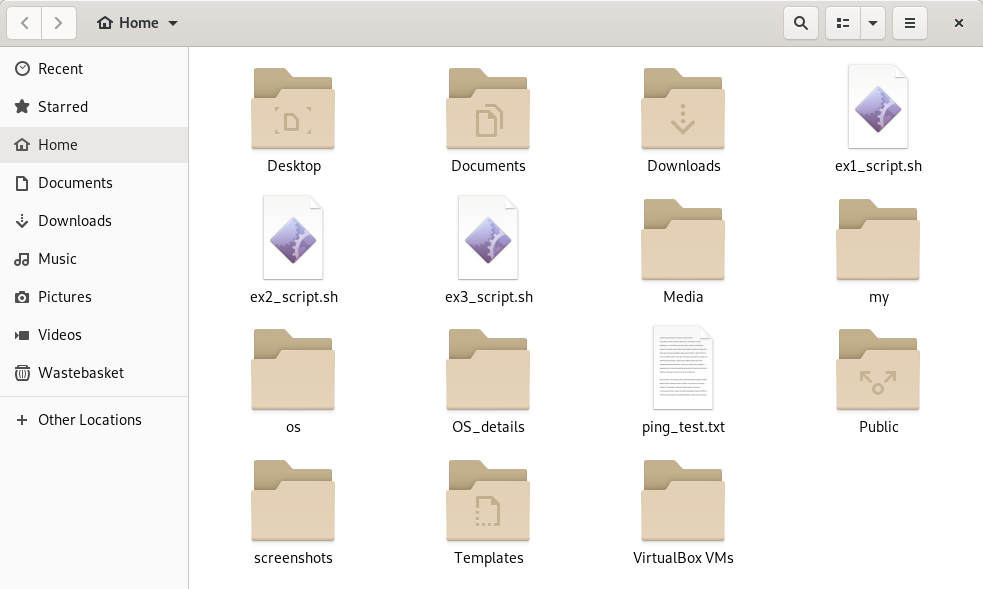
free -h | awk 'FNR == 2 {print $7}' >> OS\_details/info.txt #writes RAM available to text file

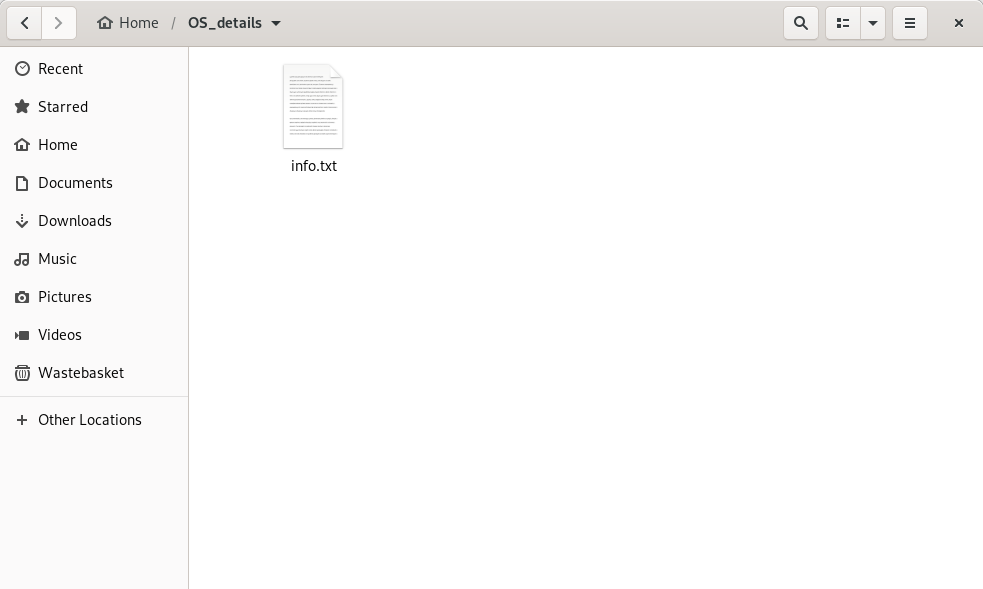
free -h | awk 'FNR == 2 {print $3}' >> OS\_details/info.txt #writes RAM used to text file

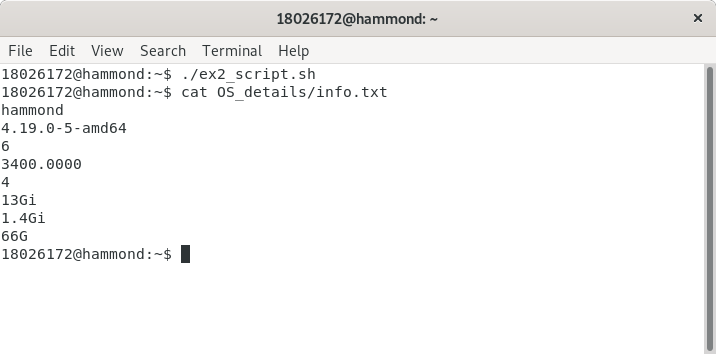
df -H / | awk 'FNR == 2 {print $3}' >> OS\_details/info.txt #writes Hard Disk space used mounted on / to text file

*Screenshots*









Exercise 3: Extracting System Information for Three Network Nodes

*Bash Script Code (ex3\_script.sh):*

#!/bin/bash

mkdir nodes\_details

ssh-keygen #generates ssh key

j=0 #counter used for text file naming

for i in {19..16} #loops through nodes

do

((j=j+1)) #increment counter

ssh-copy-id 18026172@149.170.10.$i #copy ssh key to other nodes, this is so password only has to be entered once

ssh 18026172@149.170.10.$i hostname >> nodes\_details/node${j}.txt #ssh into current node and write hostname to the nodes text file

ping -c 1 149.170.10.$i #ping current node

if [ $? -eq 0 ]; then #checks return value of ping command if code '0' then success else no reply or errors.

echo "node $j (149.170.10.$i) is on" >> nodes\_details/node${j}.txt #writes status to text file

else

echo "node $j (149.170.10.$i) is off" >> nodes\_details/node${j}.txt

fi

ssh 18026172@149.170.10.$i uname -r >> nodes\_details/node${j}.txt #ssh into current node and write kernel version and architecture to the nodes text file

ssh 18026172@149.170.10.$i lscpu | grep 'CPU family' | awk '{print $3}' >> nodes\_details/node${j}.txt #ssh into current node and write CPU family to the nodes text file

ssh 18026172@149.170.10.$i lscpu | grep 'CPU max MHz' | awk '{print $4}' >> nodes\_details/node${j}.txt

ssh 18026172@149.170.10.$i lscpu | grep 'Core(s) per socket' | awk '{print $4}' >> nodes\_details/node${j}.txt

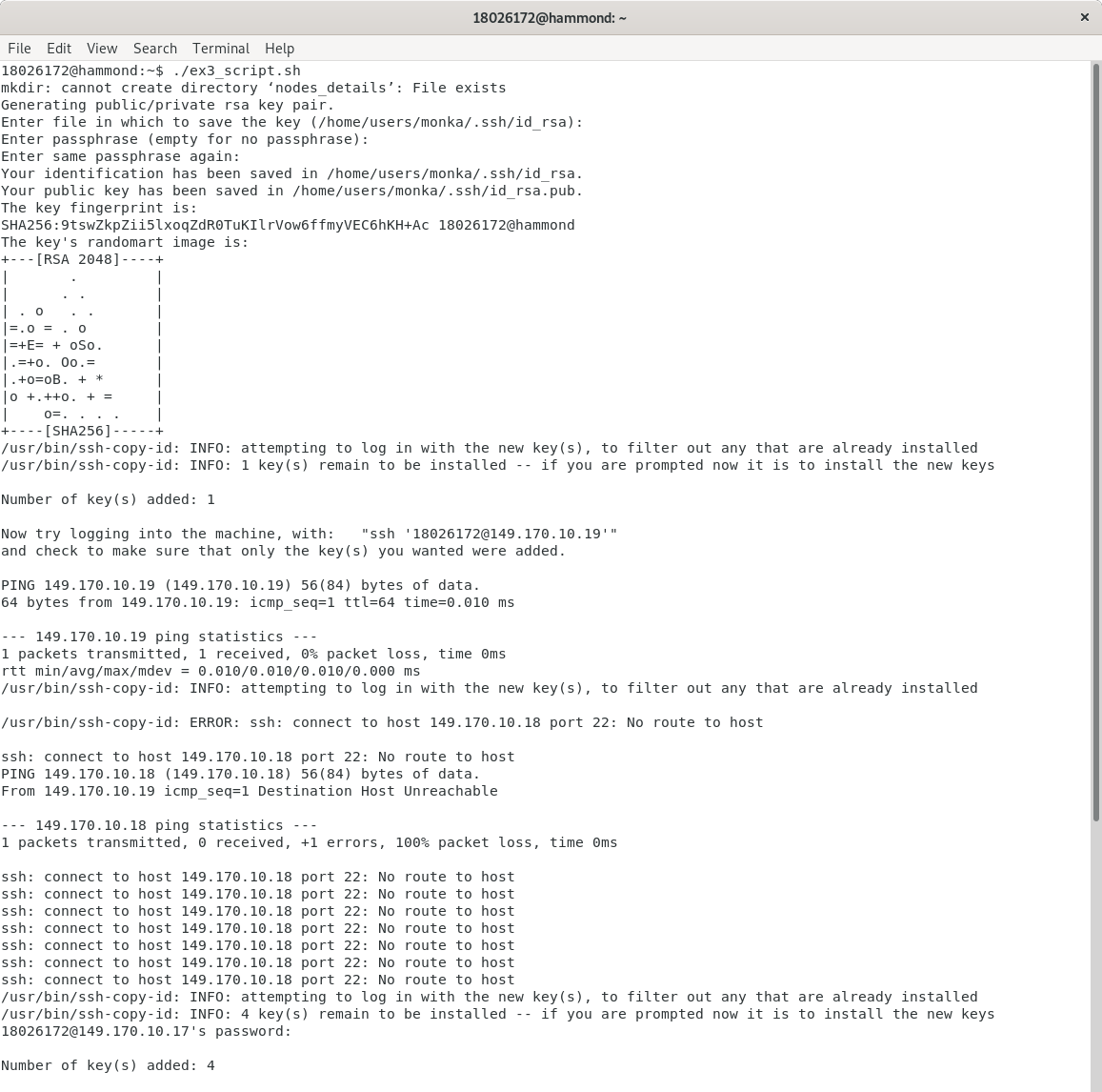
ssh 18026172@149.170.10.$i free -h | awk 'FNR == 2 {print $7}' >> nodes\_details/node${j}.txt #ssh into current node and write RAM available to text file

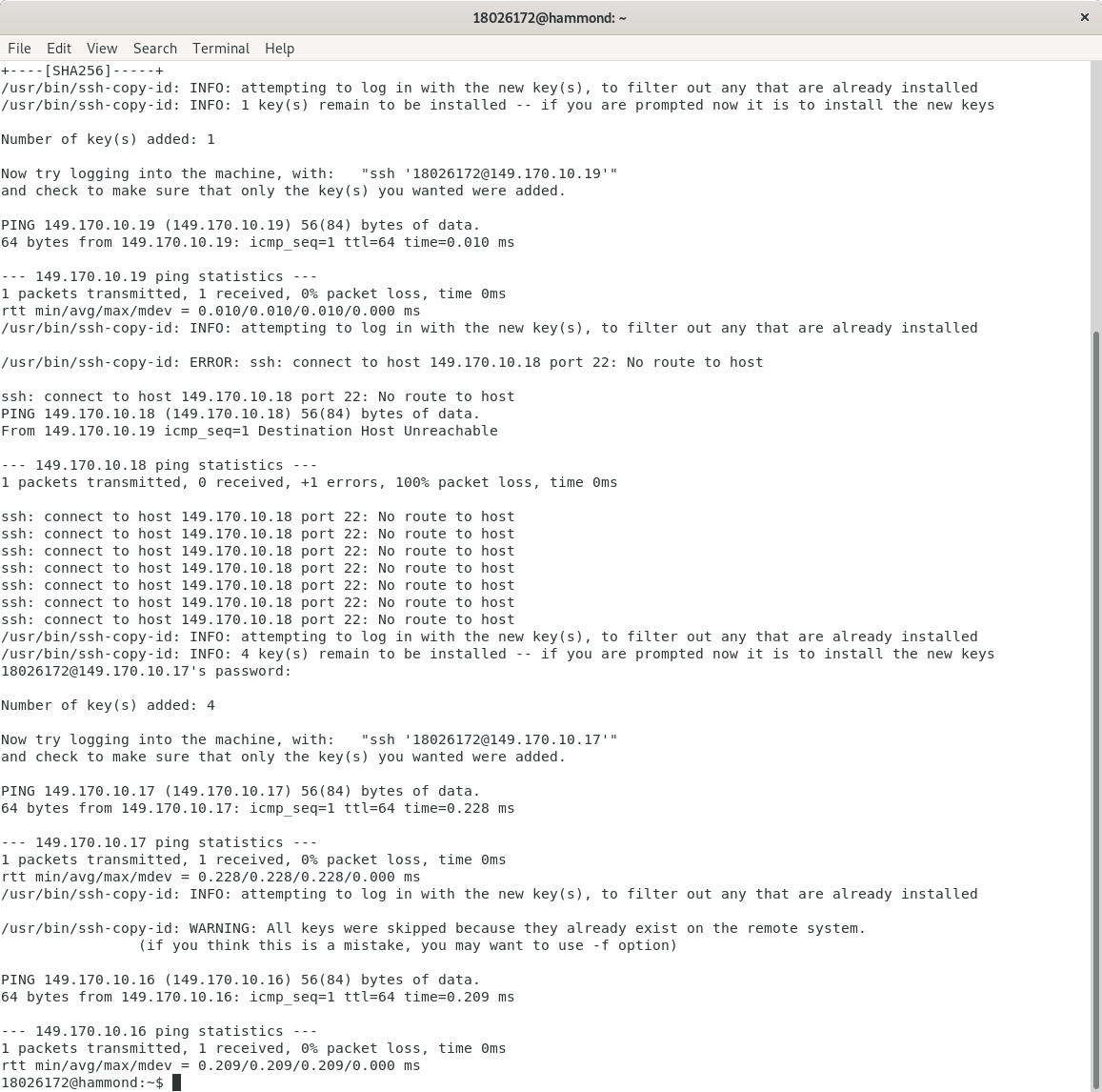
ssh 18026172@149.170.10.$i free -h | awk 'FNR == 2 {print $3}' >> nodes\_details/node${j}.txt

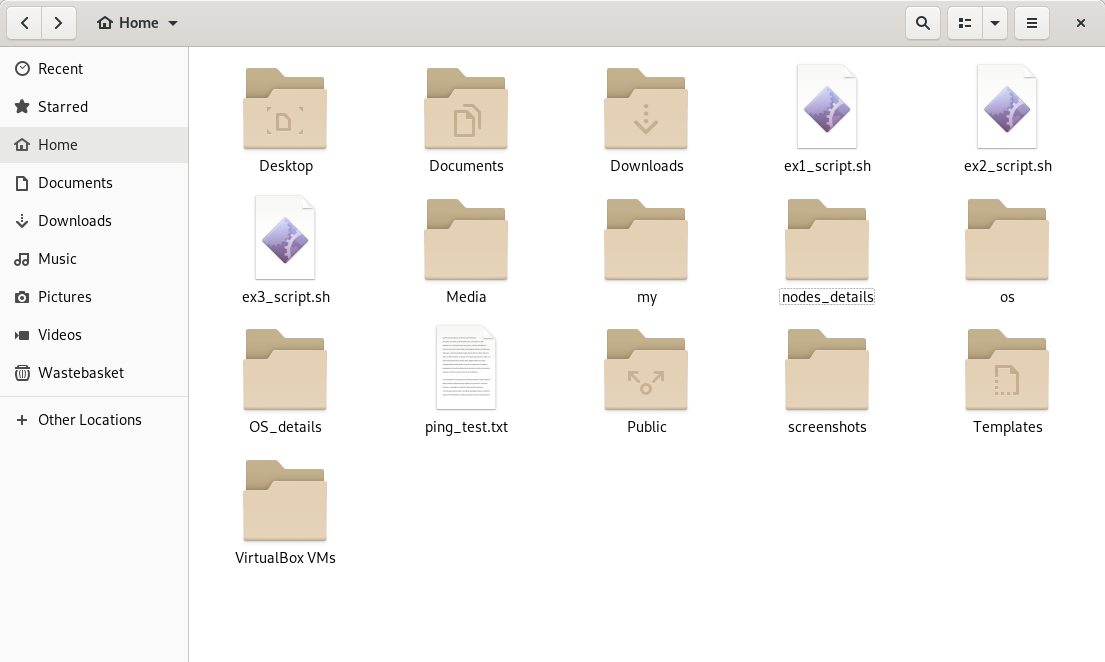
ssh 18026172@149.170.10.$i df -H / | awk 'FNR == 2 {print $3}' >> nodes\_details/node${j}.txt

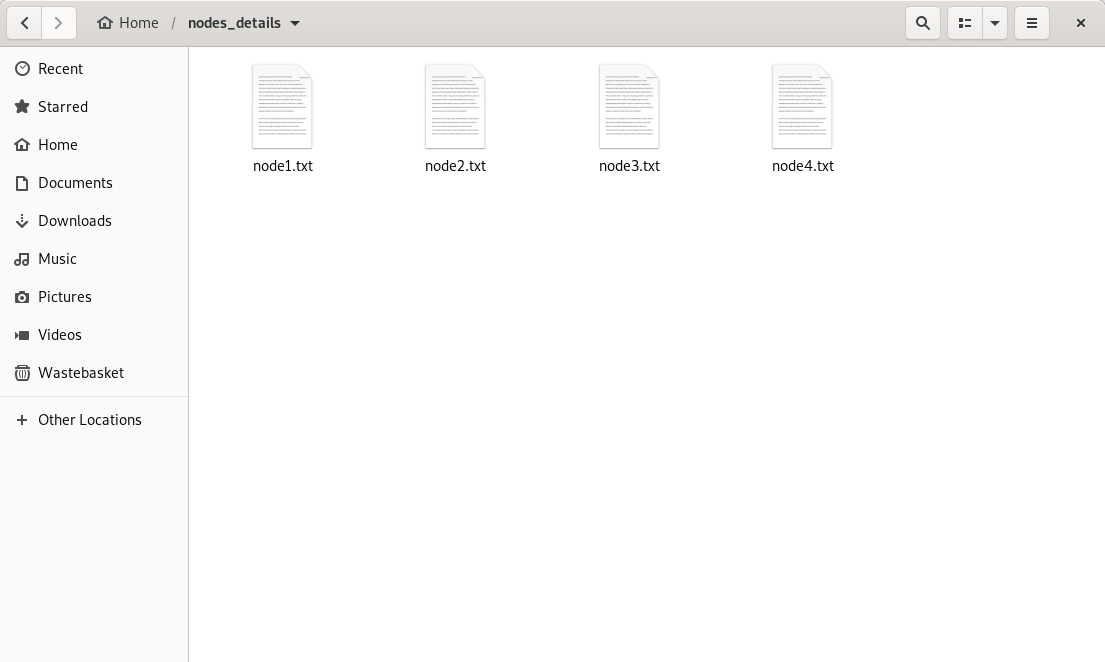
done

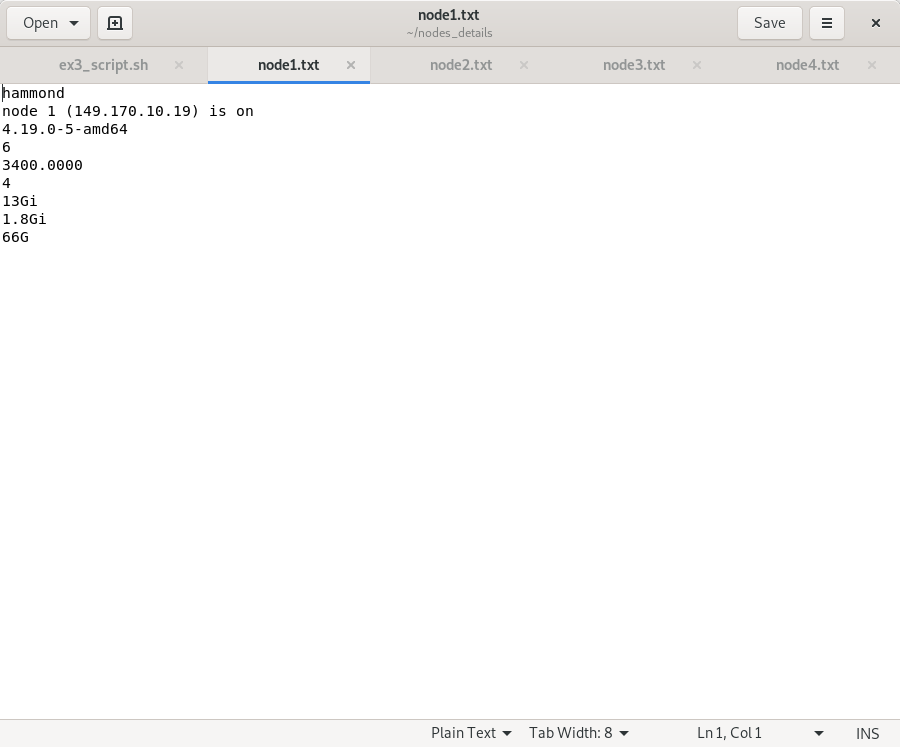
*Screenshots*

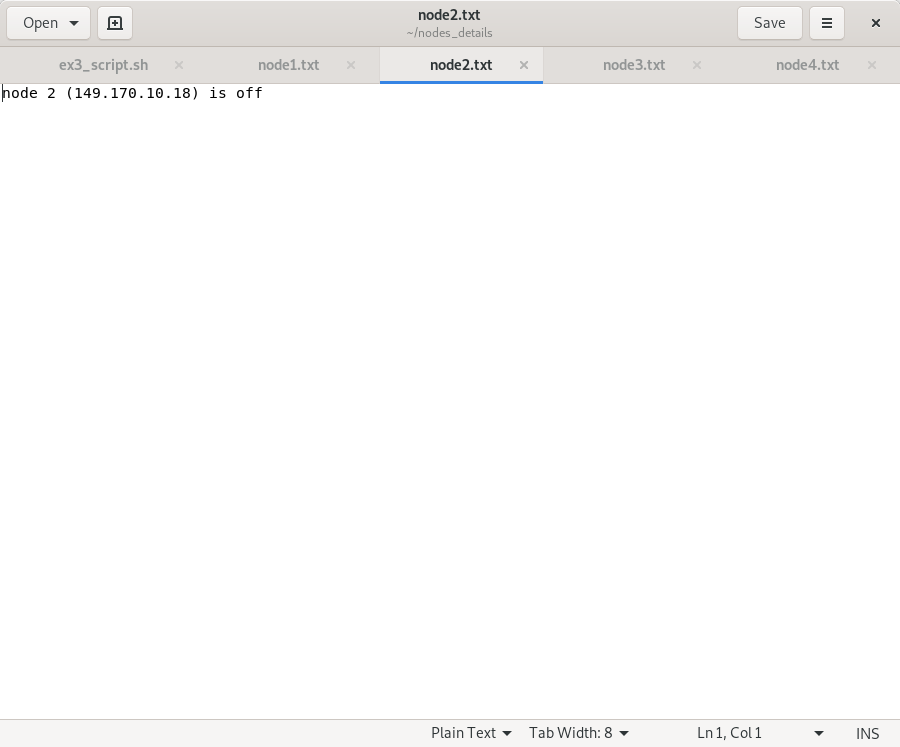


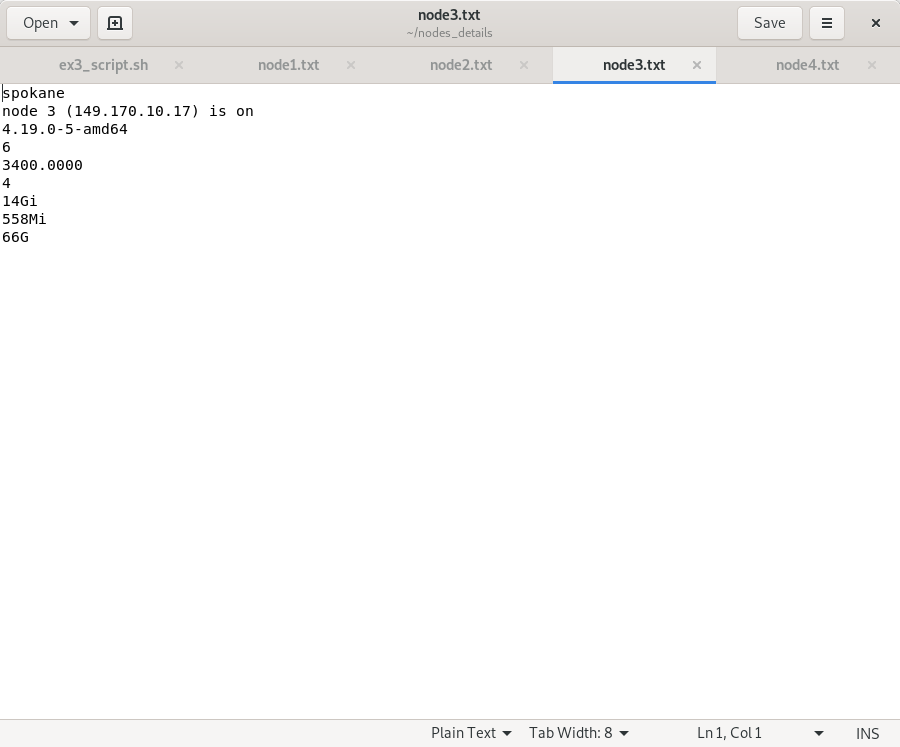


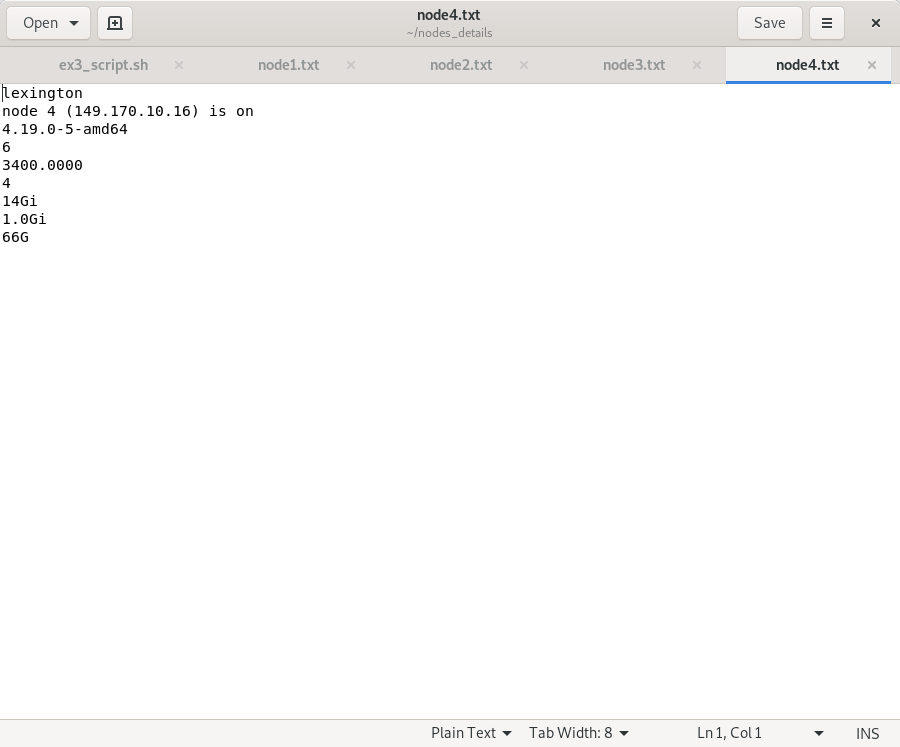






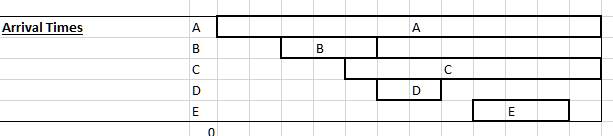




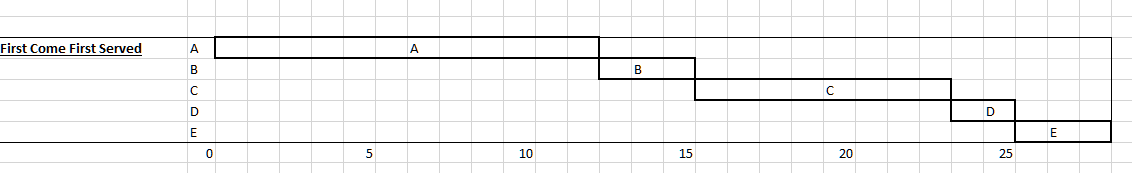


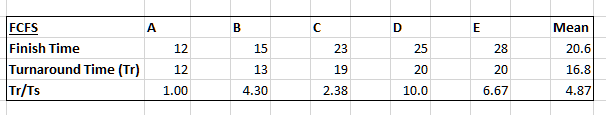
**PART C: Exercise and Reflection on Scheduling Algorithm Optimisation**

Process Scheduling Algorithms – Diagrams

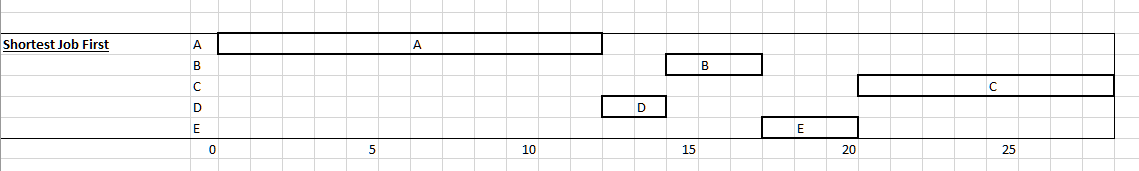


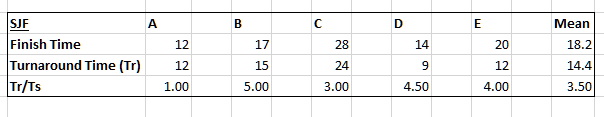
*First Come First Served (FCFS)*



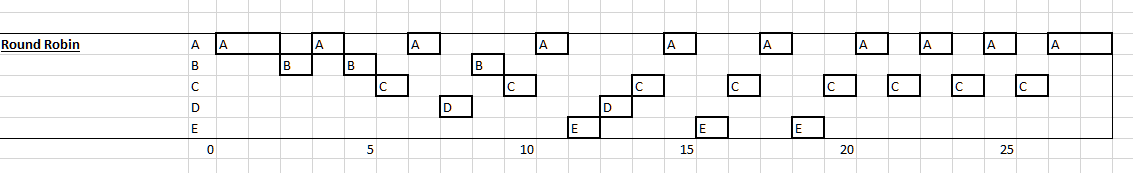


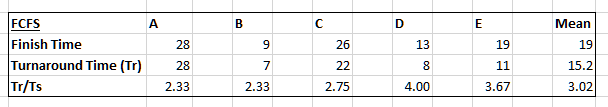
*Shortest Job First (SJF)*





*Round Robin (RR)*





Process Scheduling Algorithms - Reflection

First Come First Served is the simplest of the three scheduling algorithms, as soon as a process arrives it is added to the ready queue. Once the current process is finished the next process to get the CPU will be the one that has been in the ready queue the longest. This algorithm is much better for longer processes as opposed to short ones. With the Shortest Job First algorithm the shortest processes that are ready are executed first. However, the time taken by the process must be known by the CPU beforehand and long processes will have a greater amount of waiting time. With the Round Robin algorithm each process occupies the CPU for a fixed time quantum this means all processes are given the same priority, after a process runs for the fixed amount of time an interrupt is sent to the process and then it moves to the end of the ready queue. The throughput of the RR algorithm depends on how big the allocated time quantum is, if it’s too large the algorithm won’t behave properly and if it’s too small a lot of time will be spent switching process. For the data that has been provided the Shortest Job First algorithm is the most efficient as it has the lowest mean turnaround time (14.4) which means it has the shortest total amount of time spent waiting and executing. On the other hand, the Round Robin algorithm has the shortest mean response ratio, this means that the separate processes are seen to quicker by the CPU than the other algorithms, this could mean that Round Robin is better from the users point of view as the processes could potentially be producing an output while still being processed.

**References**

Stallings, W. (2012) *Operating Systems Internals and Design Principles*. Seventh Edition. England: Pearson Education.

Thurrott, P. (2014) *Microsoft Confirms that Windows 10 will also be Version 10 Internally*. Informa Tech. [Online] [Accessed on 30/03/2020]

<https://www.itprotoday.com/windows-server/microsoft-confirms-windows-10-will-also-be-version-10-internally>

Linus, T. (2019) *Linux 5.0*. Indiana University. [Online] [Mailing List] [Accessed on 30/03/2020]

<http://lkml.iu.edu/hypermail/linux/kernel/1903.0/01288.html>

Microsoft. (2017) *Minimum hardware requirements*. [Online] [Accessed on 30/03/2020]

<https://docs.microsoft.com/en-us/windows-hardware/design/minimum/minimum-hardware-requirements-overview>

Wikipedia. (2019) *List of Linux-supported computer architectures.* [Online] [Accessed on 30/03/2020]

<https://en.wikipedia.org/wiki/List_of_Linux-supported_computer_architectures>

Wikipedia. (2020) *Windows 10 Mobile.* [Online] [Accessed on 30/03/2020]

<https://en.wikipedia.org/wiki/Windows_10_Mobile>

Wikipedia. (2020) *Android.* [Online] [Accessed on 30/03/2020]

<https://en.wikipedia.org/wiki/Android_(operating_system)>

Hoffman, C. (2016) *What Does “Everything Is a File” Mean in Linux.* How-To Geek. [Online] [Accessed on 31/03/2020]

<https://www.howtogeek.com/117939/htg-explains-what-everything-is-a-file-means-on-linux/>

Panwar, R. (2012) *Linux File System and Windows File System, Difference*. Linux Explore. [Online] [Accessed on 31/03/2020]

<https://linuxexplore.com/2012/10/01/Linux-file-system-and-windows-file-system-difference/>

McFarlane, N. (2004) *Comparing Windows and Linux: Systematic file storage.* Tech Target. [Online] [Accessed on 31/03/2020]

<https://searchdatacenter.techtarget.com/tip/Comparing-Windows-and-Linux-Systematic-file-storage>

Kath, R. (1993) *Managing Virtual Memory*. Microsoft [Online] [Accessed on 31/03/2020]

<https://docs.microsoft.com/en-us/previous-versions/ms810627(v=msdn.10)?redirectedfrom=MSDN>

Essays, UK. (2018). *Compare the memory management of windows with Linux.* [Online] [Accessed on 31/03/2020]

<https://www.ukessays.com/essays/engineering/compare-the-memory-management.php?vref=1>

Locutus (2009) *Comparing windows and Linux hardware management*. Toolbox [Online] [Accessed on 31/03/2020]

<https://it.toolbox.com/blogs/locutus/comparing-windows-and-linux-hardware-management-090209>

Wikipedia (2020). *Usage share of operating systems.* [Online] [Accessed on 31/03/2020]

<https://en.wikipedia.org/wiki/Usage_share_of_operating_systems>

Nexsols, Inc. (2016) *Security Features Linux vs Windows.* Medium[Online] [Accessed on 31/03/2020]

<https://medium.com/@NexsolsInc./security-features-linux-vs-windows-d88666de6550>