LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CS5001NA Networks and Operating System

Assessment Weightage & Type

20% Individual Coursework

Year and Semester

2018-19 Autumn

Student Name: Abiral Joshi Shrestha

London Met ID: 17030696

College ID: NP01NT4A170008

Assignment Due Date: 15th April, 2019

Assignment Submission Date: 15th April 14, 2019

Word Count (Where Required): 1436

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a marks of zero will be awarded.

Abstract

The following project is a report on a script developed in the environment of a well-known UNIX Shell. A script that implements interaction with the UNIX environment in friendly manner and executes simple input/output operations was developed.

The second part of the project is a technical report that focuses on connection/communication of various input and output devices.

Contents

TASK A

1. Introduction

UNIX is an operating system, a stable, multi-user, multi-tasking system for servers, desktops and laptops, which was first introduced in the 1960s and has been under constant development since its launch. It has a Graphical User Interface (GUI) which makes it very simple to operate in the environment (Stonebank, 2000). Being an OS developed in 1960s, it predates the concept of personal computers, thus its multi-user, shared, networked operating environment.

There are three main shells of the UNIX OS; the Bourne shell, the C shell and the Korn shell. The Bourne Again Shell (bash) is the one used most by UNIX users (Quigley, 2000). It is the default shell of UNIX. A shell is a command interpreter. Commands can be bundled together into a script, which a shell executes. The way the commands are manipulated makes up the scripting language of the Shell.

2. Body of report

2.1 Script

```
#!/bin/bash
x=0
y=1
secret=1738          #Setting the secret key.
while [ $x == 0 ]
do
        echo -e "Enter the secret key:- \c"
        read key
        if [ $secret == $key ]
        then
        echo "The secret key has matched."
        x=1
        else
                if [ $y -lt 3 ]
```

```
                then
                        echo "The secret key didn't match, try again."
                        ((y++))
                else
                        echo "Program has been terminated."
                        exit
                        break
                fi
        fi
done
echo "Welcome, the program has been initialized." $1 $2
echo "Program initialized on:- $(date)"              #Displays time and
date
choice=y
while [ $choice == "y" ]
do
        echo "Country | Codes"              #Inputting the country
codes
        echo "Japan | JPN"
        echo "Argentina | ARG"
        echo "Germany | GER"
        echo "France | FRA"
        echo "Brazil | BRZ"

while [ True ]
do
        echo -e "Guess the best football team:- \c"
        read code
        case $code in
        "JPN") echo "Japan is the most successful nation in the
World Cup tournaments having qualified for six tournaments so far.
```

```
They are also the most successful team in terms of Asian continental competitions."
        break
        ;;
        "ARG") echo "Argentina is one of the most successful teams from the South American continent, having won the tournament twice and the Copa America 14 times."
        break
        ;;
        "GER") echo "Germany is one of the most successful teams in the World Cup, having qualified for 19 out of 21 tournaments held and winning four world titles."
        break
        ;;
        "FRA") echo "France is a major player at the FIFA World Cup, having won 2 titles, including the 2018 tournament."
        break
        ;;
        "BRZ") echo "Brazil is the most successful team in the FIFA World Cup, having participated in all tournaments and winning five titles."
        break
        ;;
        *) echo "You have entered an invalid code, try again!"
esac
done
echo "Player | Code"                    #Setting the player's codes
echo "Kagawa | KAG"
echo "Messi | MES"
echo "Reus | REU"
echo "Lloris | LOR"
```

```
echo "Coutinho | COU"
while [ true ]
do
        echo -e "Enter three codes for players: \c"
        read player_1 player_2 player_3
        echo -e "The players selected by you are: \c"
        echo -e "1. $player_1"
        echo -e "2. $player_2"
        echo -e "3. $player_3"

        echo -e "Select one player: \c"
        read player_code

        case $player_code in
        "KAG") echo -e "Shinji Kagawa was selected."
                cat Kagawa.txt                  #Importing the .txt files
        break
        ;;
        "MES") echo -e "Lionel Messi was selected."
                cat Messi.txt
        break
        ;;
        "COU") echo -e "Philippe Countinho was selected."
                cat Coutinho.txt
        break
        ;;
        "REU") echo -e "Marco Reus was selected."
                cat Reus.txt
        break
         ;;
        "LOR") echo -r "Hugo Lloris was selected."
```

```
                cat Lloris.txt
        break
        ;;
        *) echo -e "The code was invalid, try again: \c"
        esac
done

echo -e "Do you want to restart the program? (y/n) \c" #Command
                                                        to restart the
                                                        program
read choice
done
```

2.2 Testing

2.2.1 Test 1

This test was carried out in order to conclude that the program works and displays the name and time.

Test ID	1
Input	bash 17030696cw2ii Abiral 17030696
Expected Result	Displays name and ID.
Actual Result	The program welcomed the user displaying the name and ID.

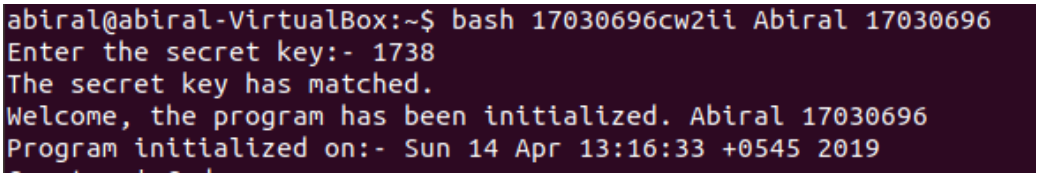

Figure 1 Test 1

2.2.2 Test 2

This test was carried to check for the description of Brazil.

Test ID	2
Input	BRZ
Expected Result	Description of Brazil.
Actual Result	Short description of Brazil.

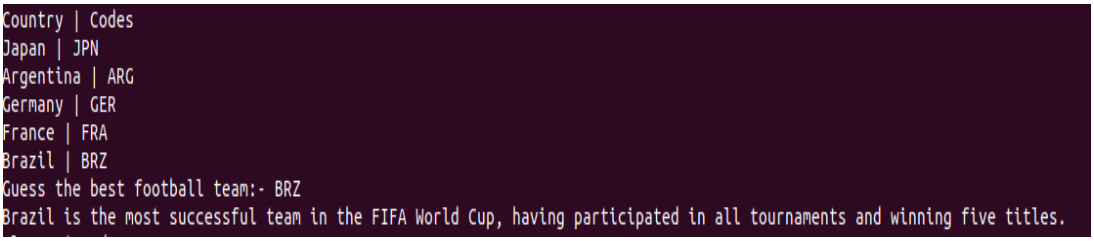

Figure 2 Test 2

2.2.3 Test 3

Test for the selection of codes of three players.

Test ID	3
Input	MES KAG COU
Expected Result	Selected codes will be displayed.
Actual Result	Selected codes are displayed.

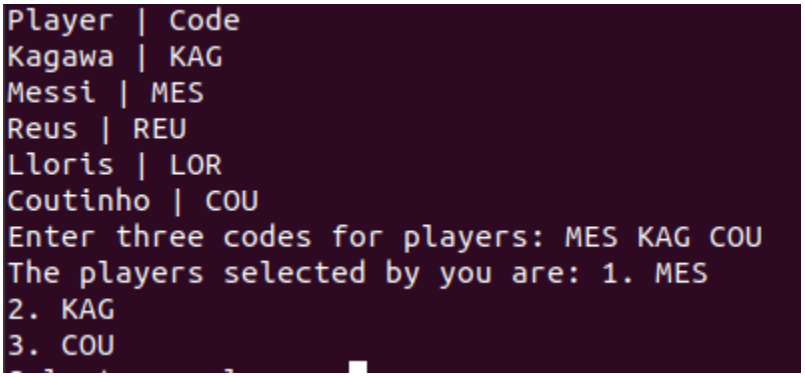

Figure 3 Test 3

2.2.4 Test 4

This test is conducted to test for program termination.

Test ID	4
Input	n
Expected Result	Program will be terminated.
Actual Result	Program was terminated.

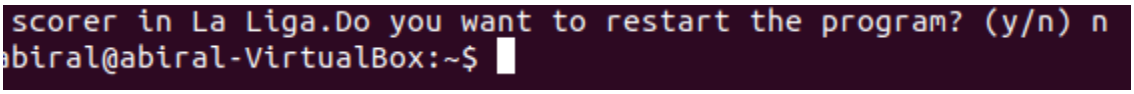

Figure 4 Test 4

3. Conclusions

The Task A of the coursework was successfully done in a script file named 17030696CW2ii. In addition, a few tests were conducted to check the proper functioning of the program.

TASK B

1. Introduction

UNIX is an operating system, a stable, multi-user, multi-tasking system for servers, desktops and laptops, which was first introduced in the 1960s and has been under constant development since its launch. It has a Graphical User Interface (GUI) which makes it very simple to operate in the environment (Stonebank, 2000). Being an OS developed in 1960s, it predates the concept of personal computers, thus its multi-user, shared, networked operating environment.

UNIX can be used as a powerful server operating system which can be used in peer to peer and client/server networks. It has various utilities which allow it to be used in a networked and distributed environment. Some of them are: ping, FTP, telnet, finger, etc. The ping utility is used to check if the remote host is responding well or not. It sends an echo request to the available hosts on the network. This utility is useful for tacking and isolating hardware and software problems, determining the status of the network and various foreign host and for testing, measuring and managing networks. The FTP (File Transfer Protocol) utility is used to upload and download a user's file from one computer to another computer. It is useful for performing tasks such as- connecting and login to a remote host, navigating directories, listing directory contents, putting and getting files and transferring files such as ASCII, EBCDIC or binary. The Telnet utility is the one which allows a computer user at one place to make a connection, login and operate on a computer at another place. The finger utility is used to display information about users on a particular host, be it local or remote (Unix / Linux - Network Communication Utilities, n.d.).

2. Aims and Objectives

In this technical report, we will be discussing about the categories of I/O devices along with the management of I/O system such as I/O subsystem. Also, the report will contain information about device drivers and device controllers.

The main focus of this report will be on the topic of connection/communication. I have referenced a few mixture of sources such as: books, journals, web resources, etc. This report is useful to those individuals who want to gain brief knowledge on the topic.

3. Body of report

3.1 Background

Input devices provide input to a computing system about the tasks it needs to perform while output devices are meant to give out results of the tasks and to establish communication between the computing system and the user. Keyboard, mouse, scanners, joysticks, touchscreens, etc. are examples of input devices while monitors, speakers, printers, etc. are some output devices (Input/Output Device (I/O Device), n.d.).

I/O devices are hardware components that allow a user to communicate with a computing system. Most I/O devices are bi-directional so they are generally characterized under storage or communications

3.2 Categories of I/O Devices

The categories of I/O devices are: block devices, character devices, file systems and network devices.

Devices that can host a file system are known as block devices. They provide a structured access to the underlying hardware. They exhibit persistence of data and support addressable block-oriented I/O. Block I/O helps us catch the frequently used blocks in memory.

Character devices provide unstructured access to the nderlying hardware. Traditional character stream devices, such as- modem, terminal multiplexer, mouse, scanner, etc. frame buffer and sound devices are all examples of character devices. Input from such devices are taken in two sensible ways: line mode and raw mode.

A file system is a software driver in the operating system which maps between low level and high level data structures. The file system of I/O devices is responsible for determining how data is managed on a block

device in order to present higher level software with a hierarchy of directories and files, including access permissions for both.

Network devices are a packet-oriented rather than being stream-oriented devices. Although they are not visible in the file system, they remain accessible through the socket interface. Network devices can be either hardware or software (Krzyzanowski, 2012).

3.3 Device Drivers and Device Controllers

Device Drivers are software pieces that tell the operating system and other software how to communicate with the various hardware components. A software program provides information to a driver to elaborate what it wants a certain component of hardware to do. Due to the use of device drivers, most software programs do not need to know how to work with the hardware components directly. In order to carry out the tasks, the program and driver only need to know how to interact with each other (What is a device driver?, n.d.).

A device controller is a hardware unit which is attached to the I/O bus of the computer and provides a hardware interface between the computer and the I/O devices. It can usually control multiple I/O devices. Most device controllers have the capability to directly read and write memory in the system. If any controller doesn't have this ability, the processor is responsible for storing the data in or reading it from the memory. This process is time consuming so all device controllers today have DMA (Direct Memory Access) capability (Device Controllers, n.d.).

A device controller and a device driver is used by very device in order to communicate with the operating system. That being said, there are differences between the two. A device driver is a computer program that controls a specific type of device attached to a computer system whereas, a device controller is a hardware part of the computer system that manages the signals going to and coming from the CPU.

3.4 I/O subsystem

The I/O subsystem is an architecture which deals with the organization of processing various tasks within the devices, controllers, channels and I/O processors. The I/O subsystem of a computer is the interface through which the system connects to the outside world (Larraza-Mendiluze, 2013).

The I/O subsystem provides various services in regards with the I/O. Some of those services are: I/O scheduling, buffering, caching, error handling, I/O protection, spooling and device reservation. Additionally, the I/O subsystem is also responsible for protecting itself from malicious users and misbehaving processes.

i) I/O Scheduling

Determining a good order to execute the request is scheduling a set of I/O requests. This helps to reduce the average waiting time, response time, turnaround time for the I/O requests to be completed. Scheduling is implemented by maintaining a queue of the requests for each device by the OS developers. The request is placed in the queue for a certain device when the application issues a blocking I/O system call. The scheduler then rearranges the order in order to improve the efficiency of the system (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

ii) Buffering

Buffer is a memory area where the data is stored for being transferred between two devices or between a device and an application. Buffering is done to manage the speed mismatch between the producer and consumer of a data stream. It also enables the support copy semantics for the application (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

iii) Caching

Cache is a part of memory where the copy of data is held. Cached copy is much easier and faster to access than the original file. The primary difference between buffering and caching is that cache holds a copy on faster storage of an item residing elsewhere whereas buffer hold onto the existing copy of the item (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

iv) Spooling and Device Reservation

A spool is a buffer that holds the output of a device that cannot accept an interleaved data stream. At a given time, several applications may want to use a single device, causing output mixes. The OS prevents this from happening (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

v) Error Handling

An OS can protect against many kinds of hardware and application errors (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

vi) I/O Protection

A user may try to disrupt the normal function of a system by issuing illegal I/O instructions. All I/O instructions are to be privileged instructions to prevent illegal I/O access (Operating System | Kernel I/O Subsystem (I/O System), n.d.).

4. Conclusion

From the above technical report, we can conclude that UNIX as a network operating system having a ton of features to carry out networking processes with other devices. It uses device drivers and device controllers to manage the various I/O devices. We also learnt about how I/O subsystems can help in many fields such as: I/O scheduling, buffering, caching, error handling, I/O protection, spooling and device reservation.

References

Device Controllers. (n.d.). Retrieved from http://www.idc-online.com/technical_references/pdfs/information_technology/Device_Controllers_Memory_Mapped_and_Port_Mapped.pdf

Input/Output Device (I/O Device). (n.d.). Retrieved from Techopedia: https://www.techopedia.com/definition/7650/input-output-device-io-device

Krzyzanowski, P. (2012). Categories of I/O. *Devices*.

Larraza-Mendiluze, E. (2013). The computer I/O subsystem. *The Computer Input/Output*.

Operating System | Kernel I/O Subsystem (I/O System). (n.d.). Retrieved from Geeks for Geeks: https://www.geeksforgeeks.org/operating-system-kernel-i-o-subsystem-i-o-system/

Quigley, E. (2000). UNIX Shells. *LINUX Journal*.

Stonebank, M. (2000). What is UNIX? *UNIX Introduction*.

Unix / Linux - Network Communication Utilities. (n.d.). Retrieved from Tutorials Point: https://www.tutorialspoint.com/unix/unix-communication.htm

Unix / Linux - Network Communocations Utilities . (n.d.). Retrieved from Tutorials Point: https://www.tutorialspoint.com/unix/unix-communication.htm

What is a device driver? (n.d.). Retrieved from Lifewire: https://www.lifewire.com/what-is-a-device-driver-2625796