

Operating Systems Assignment

Module: Computer Architecture and Operating Systems

Submitted By: Amir Bux

ID: 2526189

List of Contents

1. Introduction.....	4
2. Key Functions of General Operating Systems.....	4
3. Use of Windows OS CLI Commands	7
4. Use of Linux OS CLI Commands	12
5. Use of Windows OS Scripts.....	15
6. Use of Linux OS Scripts.....	17
7. Conclusion	19
Bibliography (IEEE Style).....	21
Appendix	23

List of Figures

Figure 1-Process management.....	5
Figure 2-Memory management.....	6
Figure 3-File and storage management.....	6
Figure 4-Security and access control.....	7
Figure 5-Command Directory.....	8
Figure 6-Command (CD)	8
Figure 7-Command (mkdir)	9
Figure 8-Command (rmdir)	9
Figure 9-Command (rename).....	10
Figure 10-(ipconfig)	10
Figure 11-netstat.....	11
Figure 12-Systeminfo.....	11
Figure 13-tasklist	12
Figure 14-ping www.google.com	12
Figure 15-pwd	12
Figure 16-su	13
Figure 17-ls.....	13
Figure 18-cd	13
Figure 19-mkdir new	14
Figure 20-rmdir new.....	14
Figure 21-touch new.txt.....	14
Figure 22-echo hello	15
Figure 23-ipconfig.....	15
Figure 24-Whoami	15
Figure 25-number count SUM.....	16
Figure 26-voting age count output.....	17
Figure 27-Addition of two numbers	18
Figure 28-Loop Execution Output	19

1. Introduction

This report examines the core functions of general-purpose operating systems and demonstrates practical engagement with both the Windows and Linux command line environments. It begins with an overview of key operating system functions before documenting a range of Batch and Bash commands across process management, file and directory management, network testing, user management and task scheduling. The report then presents original Batch and Bash scripts that automate simple routine tasks using variables, user input, arithmetic operations, loops and conditional statements. The report closes with a reflective conclusion summarising the main findings drawn from this practical work, supported throughout by citations to established operating systems literature and official vendor documentation, and by a reference list formatted in the IEEE style.

2. Key Functions of General Operating Systems

General purpose operating systems such as Windows and Linux distributions perform a set of core functions that allow computer hardware to be shared safely among running programs and users. Four functions are described below.

Process management: This is the function through which the operating system creates, schedules and terminates the programs running on a machine. Each running program is represented internally as a process, and the kernel decides which process receives the processor at any given moment through a scheduling algorithm [1]. This matters practically because a single processor core can only execute one instruction stream at a time, yet users routinely run many applications together. Both Windows and Linux expose this function to the user through graphical monitoring tools. In Windows, Task Manager lists every active process along with its processor and memory consumption, and allows an administrator to end an unresponsive task or adjust its scheduling priority [4]. In Linux distributions such as those based on Red Hat, the equivalent information is available through the System Monitor graphical tool or through the terminal commands `ps` and `top`, which display a live table of running processes and their resource usage [5].

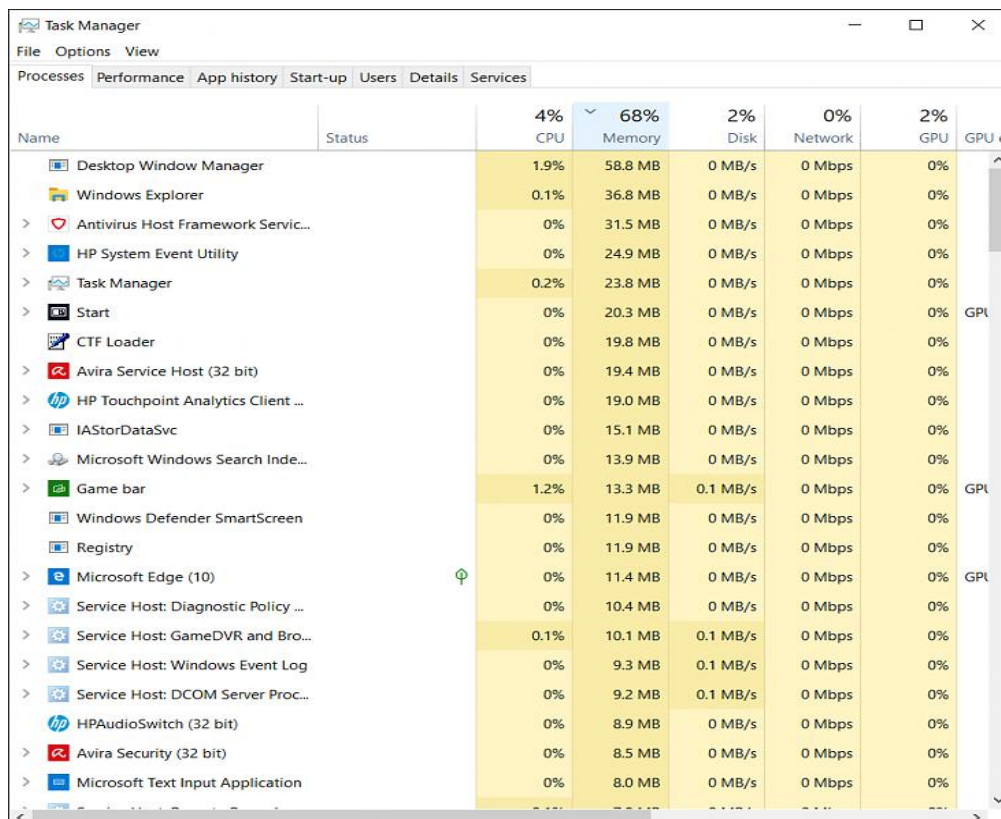


Figure 1-Process management

Memory management: This is the function responsible for allocating the limited amount of random access memory among competing processes, and for reclaiming that memory once a process ends. Modern operating systems achieve this through virtual memory, a technique that gives each process the illusion of a private, continuous address space while the kernel transparently maps that space onto physical memory and disk backed storage [2]. The Performance tab within Windows Task Manager displays current memory usage, commit charge and page file activity, giving a direct view of this function in operation. On Linux, the free command and the Memory section of System Monitor provide comparable figures.

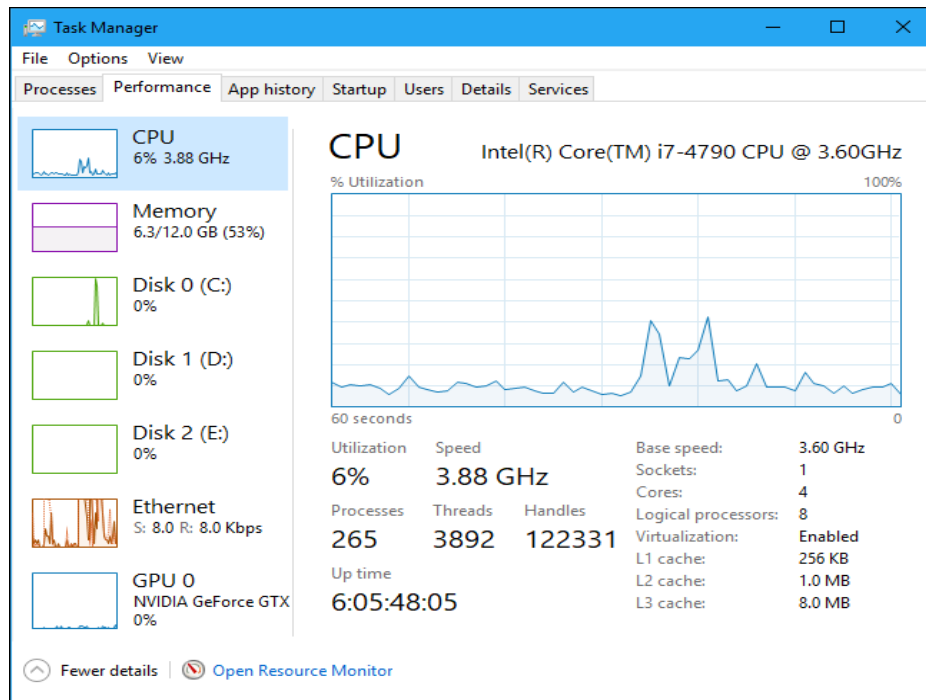


Figure 2-Memory management

File and storage management: It governs how data is organised into files and directories on a storage device, and how that data is protected from accidental loss or corruption. The operating system presents a hierarchical file system to the user while managing the underlying disk blocks, permissions and metadata behind the scenes [3]. File Explorer in Windows and the Files application or terminal commands such as ls in Linux both expose this function, allowing a user to browse, move and inspect files without needing to understand how the data is physically stored.

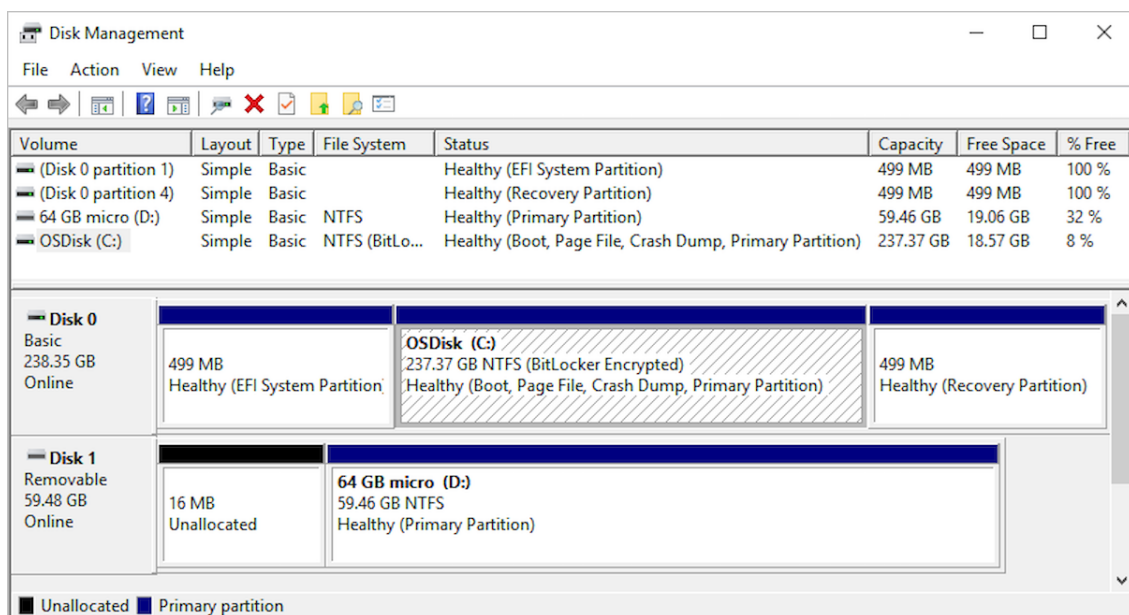


Figure 3-File and storage management

Security and access control: is the function that determines which users and processes may read, write or execute particular resources. This is enforced through user accounts, permission structures and, in many systems, mandatory access control frameworks that restrict what even a compromised process may do [1]. Windows implements this through NTFS permissions and User Account Control, while Linux implements it through owner, group and other permission bits together with tools such as chmod and chown.

Taken together, these four functions illustrate a consistent principle across otherwise different operating systems. Each system must mediate access to processor time, memory, storage and security on behalf of many competing programs, and the graphical and command line tools described above are simply the visible surface of that underlying coordination work [2], [3].

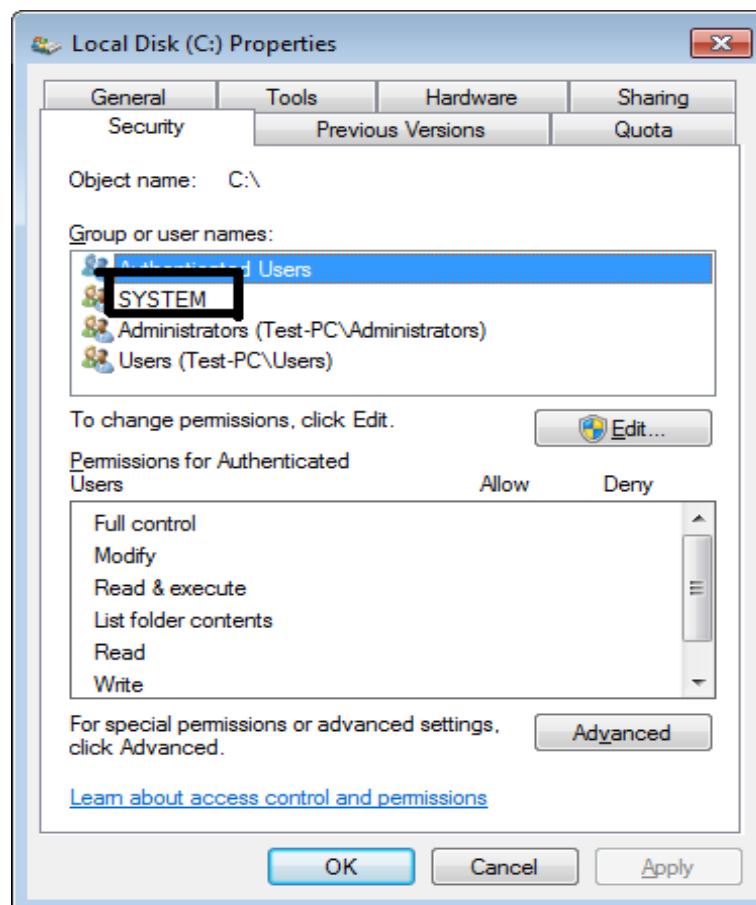


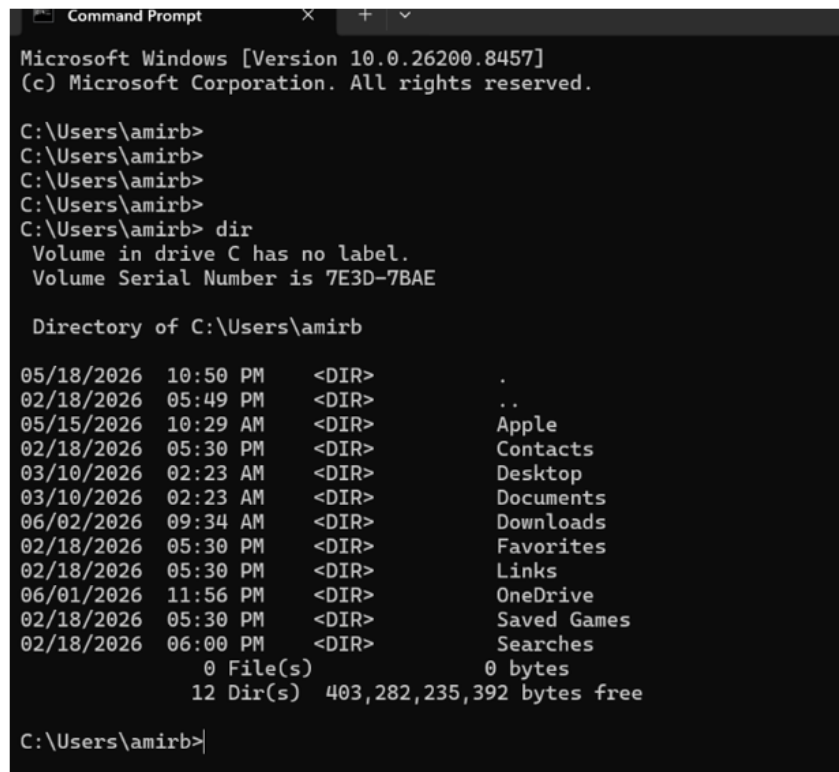
Figure 4-Security and access control

3. Use of Windows OS CLI Commands

This section documents the use of ten Windows command line utilities across process management, file and directory management, network testing, user management and task scheduling.

1. Command (Dir): The DIR command is used in the Windows Command Prompt to display the files and folders contained within a selected directory. It helps users view file names, extensions,

sizes, and modification dates, making file management and navigation easier through the command-line interface.



```
Microsoft Windows [Version 10.0.26200.8457]
(c) Microsoft Corporation. All rights reserved.

C:\Users\amirb>
C:\Users\amirb>
C:\Users\amirb>
C:\Users\amirb>
C:\Users\amirb> dir
Volume in drive C has no label.
Volume Serial Number is 7E3D-7BAE

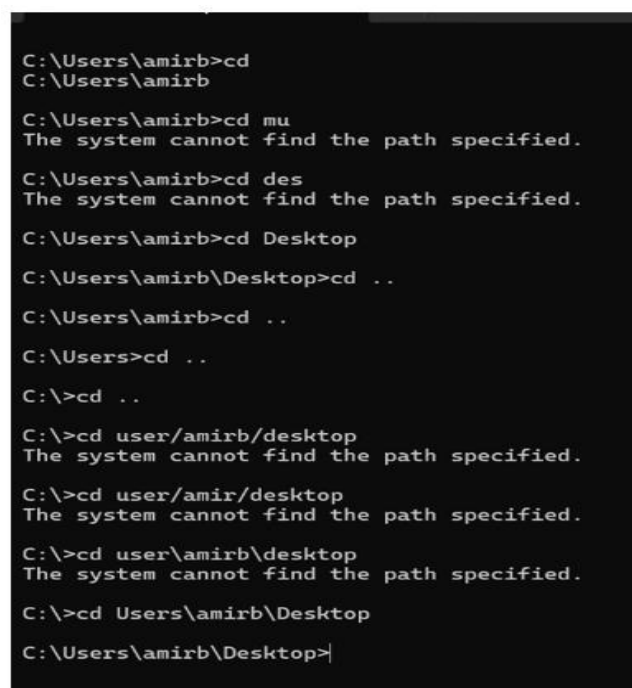
Directory of C:\Users\amirb

05/18/2026  10:50 PM    <DIR>          .
02/18/2026  05:49 PM    <DIR>          ..
05/15/2026  10:29 AM    <DIR>          Apple
02/18/2026  05:30 PM    <DIR>          Contacts
03/10/2026  02:23 AM    <DIR>          Desktop
03/10/2026  02:23 AM    <DIR>          Documents
06/02/2026  09:34 AM    <DIR>          Downloads
02/18/2026  05:30 PM    <DIR>          Favorites
02/18/2026  05:30 PM    <DIR>          Links
06/01/2026  11:56 PM    <DIR>          OneDrive
02/18/2026  05:30 PM    <DIR>          Saved Games
02/18/2026  06:00 PM    <DIR>          Searches
               0 File(s)                0 bytes
              12 Dir(s)  403,282,235,392 bytes free

C:\Users\amirb>
```

Figure 5-Command Directory

2. Command (Cd): The CD (Change Directory) command is used to navigate between folders in the Windows Command Prompt. It allows users to move to a different directory, making it easier to access, organise, and manage files efficiently.



```
C:\Users\amirb>cd
C:\Users\amirb>
C:\Users\amirb>cd mu
The system cannot find the path specified.
C:\Users\amirb>cd des
The system cannot find the path specified.
C:\Users\amirb>cd Desktop
C:\Users\amirb\Desktop>cd ..
C:\Users\amirb>cd ..
C:\Users>cd ..
C:\>cd ..
C:\>cd user/amirb/desktop
The system cannot find the path specified.
C:\>cd user/amir/desktop
The system cannot find the path specified.
C:\>cd user\amirb\desktop
The system cannot find the path specified.
C:\>cd Users\amirb\Desktop
C:\Users\amirb\Desktop>
```

Figure 6-Command (CD)

3. Command (mkdir): The MKDIR (Make Directory) command is used to create a new folder or directory in the Windows Command Prompt. It helps users organise files by creating directories for storing and managing data efficiently.

```

4 Dir(s) 401,633,783,808 bytes free

C:\Users\amirb\Desktop>mkdir folder one folder two folder three folder four
A subdirectory or file folder already exists.
Error occurred while processing: folder.
A subdirectory or file folder already exists.
Error occurred while processing: folder.
A subdirectory or file folder already exists.
Error occurred while processing: folder.

C:\Users\amirb\Desktop>mkdir folder01

C:\Users\amirb\Desktop>
C:\Users\amirb\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is 7E3D-7BAE

Directory of C:\Users\amirb\Desktop

06/02/2026 11:28 AM <DIR> .
05/18/2026 10:50 PM <DIR> ..
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> four
06/02/2026 11:19 AM <DIR> new
06/02/2026 11:27 AM <DIR> one
06/02/2026 11:27 AM <DIR> three
06/02/2026 11:27 AM <DIR> two
0 File(s) 0 bytes
10 Dir(s) 401,634,680,832 bytes free

C:\Users\amirb\Desktop>mkdir |

```

Figure 7-Command (mkdir)

4. Command (rmdir): The rmdir command is used in the Windows Command Prompt to delete an empty directory or folder from the file system. It helps remove unwanted directories efficiently.

```

Directory of C:\Users\amirb\Desktop

06/02/2026 11:28 AM <DIR> .
05/18/2026 10:50 PM <DIR> ..
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> four
06/02/2026 11:19 AM <DIR> new
06/02/2026 11:27 AM <DIR> one
06/02/2026 11:27 AM <DIR> three
06/02/2026 11:27 AM <DIR> two
0 File(s) 0 bytes
10 Dir(s) 401,628,364,800 bytes free

C:\Users\amirb\Desktop>rmdir new one three two

C:\Users\amirb\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is 7E3D-7BAE

Directory of C:\Users\amirb\Desktop

06/02/2026 11:35 AM <DIR> .
05/18/2026 10:50 PM <DIR> ..
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> four
0 File(s) 0 bytes
6 Dir(s) 401,628,471,296 bytes free

C:\Users\amirb\Desktop>

```

Figure 8-Command (rmdir)

5. Command (rename): The rename command is used in the Windows Command Prompt to change the name of a file or directory without changing its location or contents.

```

Command Prompt
06/02/2026 11:44 AM <DIR> .
05/18/2026 10:50 PM <DIR> ..
06/02/2026 11:42 AM <DIR> 1
06/02/2026 11:42 AM <DIR> 100
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> four
06/02/2026 11:44 AM 0 rename
1 File(s) 0 bytes
8 Dir(s) 401,622,552,576 bytes free

C:\Users\amirb\Desktop>rename four foldertoday

C:\Users\amirb\Desktop>dir
Volume in drive C has no label.
Volume Serial Number is 7E3D-7BAE

Directory of C:\Users\amirb\Desktop

06/02/2026 11:47 AM <DIR> .
05/18/2026 10:50 PM <DIR> ..
06/02/2026 11:42 AM <DIR> 1
06/02/2026 11:42 AM <DIR> 100
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> foldertoday
06/02/2026 11:44 AM 0 rename
1 File(s) 0 bytes
8 Dir(s) 401,622,487,040 bytes free

C:\Users\amirb\Desktop>

```

Figure 9-Command (rename)

6. Command (ipconfig): The ipconfig command displays the current IP address, subnet mask, default gateway, and other network configuration details of a Windows computer through Command Prompt.

```

06/02/2026 11:42 AM <DIR> 100
03/10/2026 02:23 AM <DIR> Course Work
06/02/2026 11:27 AM <DIR> folder
06/02/2026 11:28 AM <DIR> folder01
06/02/2026 11:27 AM <DIR> foldertoday
06/02/2026 11:44 AM 0 rename
1 File(s) 0 bytes
8 Dir(s) 401,622,487,040 bytes free

C:\Users\amirb\Desktop>ipconfig

Windows IP Configuration

Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Wi-Fi:

    Connection-specific DNS Suffix  . :
    Link-local IPv6 Address . . . . . : fe80::e89b:206c:770f:5b11%5
    IPv4 Address. . . . . : 10.78.180.150
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 10.78.180.55

Ethernet adapter Bluetooth Network Connection:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

C:\Users\amirb\Desktop>

```

Figure 10-(ipconfig)

7. Command (netstat): The netstat command displays active network connections, listening ports, routing tables, and network statistics, helping users monitor and troubleshoot network activity in Windows.

```

C:\Users\amirb\Desktop>netstat

Active Connections

Proto Local Address           Foreign Address         State
TCP    10.78.180.150:49169      52.110.3.74:https       ESTABLISHED
TCP    10.78.180.150:49417      172.187.86.73:https     ESTABLISHED
TCP    10.78.180.150:52511      yulhrs-in-f188:5228    FIN_WAIT_2
TCP    10.78.180.150:56287      yulhrs-in-f188:5228    ESTABLISHED
TCP    10.78.180.150:56288      192.168.1.103:ms-do     SYN_SENT
TCP    10.78.180.150:57484      a2-19-117-150:https     CLOSE_WAIT
TCP    10.78.180.150:57485      a2-19-117-150:https     CLOSE_WAIT
TCP    10.78.180.150:57486      a95-100-153-157:https   CLOSE_WAIT
TCP    10.78.180.150:57487      52.107.242.101:https    ESTABLISHED
TCP    10.78.180.150:57488      52.107.242.51:https     ESTABLISHED
TCP    10.78.180.150:59601      20.108.130.246:https    ESTABLISHED
TCP    10.78.180.150:59604      52.104.68.39:https      ESTABLISHED
TCP    10.78.180.150:59605      172.187.86.73:https     ESTABLISHED
TCP    10.78.180.150:59607      13.89.179.13:https      ESTABLISHED
TCP    10.78.180.150:59609      72.145.35.109:https     TIME_WAIT
TCP    10.78.180.150:59610      72.145.35.109:https     TIME_WAIT
TCP    10.78.180.150:59611      72.145.35.109:https     ESTABLISHED
TCP    10.78.180.150:59612      72.145.35.109:https     ESTABLISHED
TCP    10.78.180.150:59613      52.110.10.59:https      TIME_WAIT
TCP    10.78.180.150:59615      13.89.179.11:https      ESTABLISHED
TCP    10.78.180.150:59616      52.110.10.11:https      TIME_WAIT
TCP    10.78.180.150:59617      52.110.10.11:https      TIME_WAIT
TCP    127.0.0.1:49670         DESKTOP-8SM4PVT:49671  ESTABLISHED
TCP    127.0.0.1:49671         DESKTOP-8SM4PVT:49670  ESTABLISHED

C:\Users\amirb\Desktop>

```

Figure 11-netstat

8. Command (systeminfo): The systeminfo command displays detailed information about the windows computer, including the operating system, processor, memory, BIOS version, system uptime, and installed updates.

```

C:\Users\amirb\Desktop>systeminfo

Host Name:                  DESKTOP-8SM4PVT
OS Name:                    Microsoft Windows 11 Pro
OS Version:                 10.0.26200 N/A Build 26200
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:               Multiprocessor Free
Registered Owner:           amirbux60@gmail.com
Registered Organization:    N/A
Product ID:                 90330-53254-9959H-AAQEM
Original Install Date:      2/17/2026, 9:34:45 PM
System Boot Time:           6/1/2026, 2:30:37 PM
System Manufacturer:        Dell Inc.
System Model:                Latitude 7410
System Type:                 x64-based PC
Processor(s):                1 Processor(s) Installed.
                             [01]: Intel64 Family 6 Model 142 Stepping 12 GenuineIntel ~1803 Mhz
BIOS Version:                Dell Inc. 1.21.1, 2/22/2023
Windows Directory:          C:\WINDOWS
System Directory:            C:\WINDOWS\system32
Boot Device:                 \Device\HarddiskVolume1
System Locale:                en-us;English (United States)
Input Locale:                en-us;English (United States)
Time Zone:                   (UTC+00:00) Dublin, Edinburgh, Lisbon, London
Total Physical Memory:       32,433 MB
Available Physical Memory:   24,237 MB
Virtual Memory: Max Size:    34,481 MB
Virtual Memory: Available:   26,641 MB
Virtual Memory: In Use:       7,840 MB
Page File Location(s):       C:\pagefile.sys
Domain:                       WORKGROUP

```

Figure 12-Systeminfo

9. Command (tasklist): The tasklist command displays a list of all running processes and applications on a Windows computer, including their process IDs (PIDs) and memory usage.

```
C:\Users\amirb\Desktop>tasklist
```

Image Name	PID	Session Name	Session#	Mem Usage
System Idle Process	0	Services	0	8 K
System	4	Services	0	176 K
Secure System	140	Services	0	106,724 K
Registry	184	Services	0	56,844 K
smss.exe	608	Services	0	1,676 K
csrss.exe	1440	Services	0	7,076 K
wininit.exe	1520	Services	0	9,664 K
csrss.exe	1528	Console	1	8,272 K
winlogon.exe	1628	Console	1	18,356 K
services.exe	1644	Services	0	13,156 K
Lsass.exe	1684	Services	0	4,732 K
lsass.exe	1692	Services	0	40,864 K
svchost.exe	1828	Services	0	48,340 K
fontdrvhost.exe	1840	Console	1	9,212 K
fontdrvhost.exe	1848	Services	0	5,536 K
WUDFHost.exe	1932	Services	0	18,948 K
svchost.exe	1996	Services	0	22,292 K
svchost.exe	2036	Services	0	11,488 K
svchost.exe	2064	Services	0	7,224 K
dwm.exe	2084	Console	1	184,012 K
svchost.exe	2092	Services	0	14,512 K
svchost.exe	2100	Services	0	50,356 K
svchost.exe	2144	Services	0	11,108 K
svchost.exe	2196	Services	0	19,472 K
svchost.exe	2220	Services	0	16,904 K
svchost.exe	2332	Services	0	13,492 K
svchost.exe	2376	Services	0	16,340 K
svchost.exe	2388	Services	0	14,776 K
svchost.exe	2444	Services	0	10,528 K

Figure 13-tasklist

10. Command (ping www.google.com): The ping www.google.com command tests internet connectivity by sending data packets to Google's server and measuring the response time and packet loss.

```
C:\Users\amirb\Desktop>ping www.google.com

Pinging www.google.com [142.251.157.119] with 32 bytes of data:
Reply from 142.251.157.119: bytes=32 time=26ms TTL=114
Reply from 142.251.157.119: bytes=32 time=75ms TTL=114
Reply from 142.251.157.119: bytes=32 time=136ms TTL=114
Reply from 142.251.157.119: bytes=32 time=159ms TTL=114

Ping statistics for 142.251.157.119:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 26ms, Maximum = 159ms, Average = 99ms

C:\Users\amirb\Desktop>
```

Figure 14-ping www.google.com

4. Use of Linux OS CLI Commands

This section documents the use of ten Linux command line utilities across the same five categories, using Bash on a Linux distribution.

1. Command (pwd): The pwd command displays the current working directory, helping users identify their present location in the Linux file system through the command-line interface.

```
sysadmin@localhost:~$ pwd
/home/sysadmin
sysadmin@localhost:~$
```

Figure 15-pwd

2. Command (su): The su command switches the current user to another user account, usually the root user, after successful authentication with the appropriate password.

```
sysadmin@localhost:~$ su
Password:
root@localhost:~# su sysadmin
sysadmin@localhost:~$
```

Figure 16-su

3. Command (ls): The ls command lists files and directories in the current location. It can also display detailed information using different options and flags.

```
sysadmin@localhost:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
sysadmin@localhost:~$ ls -a
.          .bashrc    .selected_editor  Downloads  Public
..         .cache     Desktop           Music      Templates
.bash_logout .profile  Documents         Pictures   Videos
sysadmin@localhost:~$ ls -l
total 4
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Desktop
drwx----- 4 sysadmin sysadmin 4096 Dec 20 2017 Documents
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Downloads
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Music
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Pictures
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Public
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Templates
drwx----- 2 sysadmin sysadmin   6 Dec 20 2017 Videos
sysadmin@localhost:~$
```

Figure 17-ls

4. Command (cd): The cd command changes the current working directory, allowing users to navigate between different folders and locations within the Linux file system.

```
sysadmin@localhost:~$ cd Desktop/
sysadmin@localhost:~/Desktop$
```

Figure 18-cd

5. Command (mkdir new): The mkdir new command creates a new directory named new in the current working directory, helping users organise files and manage the Linux file system.

```
sysadmin@localhost:~$ cd Desktop/  
sysadmin@localhost:~/Desktop$ mkdir new  
sysadmin@localhost:~/Desktop$ ls  
new  
sysadmin@localhost:~/Desktop$
```

Figure 19-mkdir new

6. Command (rmdir new): The rmdir new command removes the empty directory named new from the current working directory. It only works if the directory contains no files or subdirectories.

```
sysadmin@localhost:~/Desktop$ rmdir new  
sysadmin@localhost:~/Desktop$ ls  
sysadmin@localhost:~/Desktop$ cd new  
bash: cd: new: No such file or directory  
sysadmin@localhost:~/Desktop$
```

Figure 20-rmdir new

7. Command (touch new.txt): The touch new.txt command creates a new empty file named new.txt in the current directory. If the file already exists, it updates its timestamp.

```
sysadmin@localhost:~/Desktop$ touch new.txt  
sysadmin@localhost:~/Desktop$ ls  
new.txt  
sysadmin@localhost:~/Desktop$
```

Figure 21-touch new.txt

8. Command (echo hello): The echo hello command displays the text "hello" on the terminal screen. It is commonly used to print messages or display command output.

```

sysadmin@localhost:~/Desktop$ echo hello >new.txt
sysadmin@localhost:~/Desktop$ cat new.txt
sysadmin@localhost:~/Desktop$ cat new.txt
sysadmin@localhost:~/Desktop$ cat new.txt
sysadmin@localhost:~/Desktop$ echo hello >new.txt
sysadmin@localhost:~/Desktop$ cat new.txt
hello
sysadmin@localhost:~/Desktop$ █

```

Figure 22-echo hello

9. Command (ifconfig): The ifconfig command displays and configures network interface information, including IP addresses, subnet masks, and interface status, helping users manage network settings in Linux.

```

sysadmin@localhost:~/Desktop$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:c0:a8:01:02
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:58 errors:0 dropped:0 overruns:0 frame:0
          TX packets:57 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:4494 (4.4 KB)  TX bytes:3678 (3.6 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:100 (100.0 B)  TX bytes:100 (100.0 B)

sysadmin@localhost:~/Desktop$ █

```

Figure 23-ipconfig

10. Command (whoami): The whoami command displays the username of the currently logged-in user, helping identify the active user account in the Linux operating system.

```

sysadmin@localhost:~/Desktop$ whoami
sysadmin
sysadmin@localhost:~/Desktop$ date
Tue Jun  9 10:46:37 UTC 2026
sysadmin@localhost:~/Desktop$

```

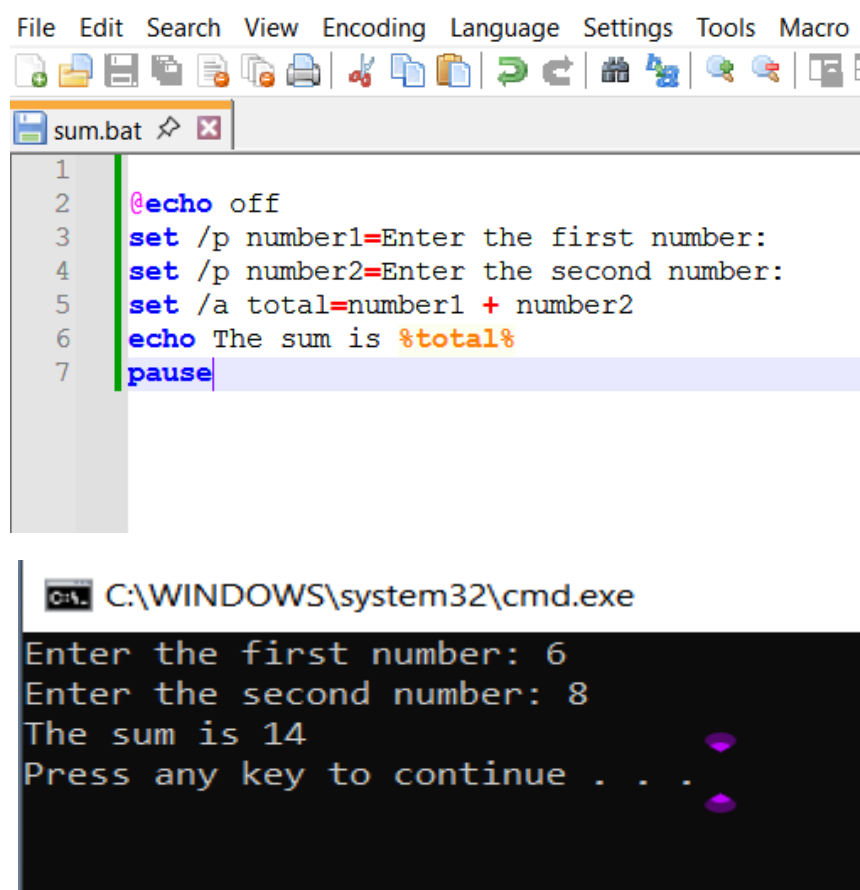
Figure 24-Whoami

5. Use of Windows OS Scripts

Windows OS scripts are used to automate system tasks, simplify administrative operations, and improve workflow efficiency. Batch scripts execute predefined commands, manage files and folders, configure system settings, and perform repetitive processes without manual intervention. They are commonly used for software installation, system maintenance, backups, user management, and task scheduling. Windows scripts help administrators reduce errors, save

time, and maintain consistent execution of operations across multiple systems. By using command-line instructions and scripting logic, users can create automated solutions for both personal and enterprise-level requirements. These scripts provide a flexible approach to managing Windows environments and enhancing overall system productivity. [15].

1. Script (number count sum): This Windows batch script is designed to calculate the sum of two numbers entered by the user. The command `@echo off` hides the commands being executed, providing a cleaner output screen. The `set /p` command is used twice to request user input and store the values in the variables `number1` and `number2`. The `set /a` command performs arithmetic calculations and adds the two stored values, saving the result in the `total` variable. The `echo` command displays the calculated sum on the Command Prompt screen using the value stored in `%total%` [23]. Finally, the `pause` command keeps the Command Prompt window open and waits for the user to press a key before closing. This script demonstrates basic input, processing, and output operations in Windows batch programming.



The image shows two parts: a script editor window and a command prompt window. The script editor window, titled 'sum.bat', contains the following code:

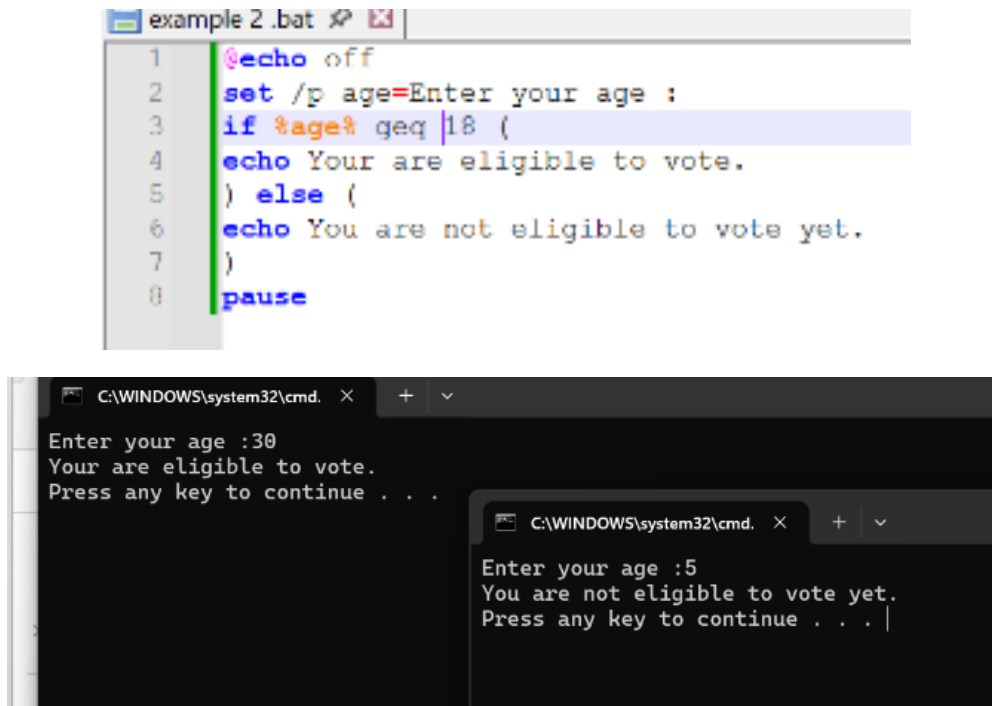
```
1
2 @echo off
3 set /p number1=Enter the first number:
4 set /p number2=Enter the second number:
5 set /a total=number1 + number2
6 echo The sum is %total%
7 pause
```

The command prompt window, titled 'C:\WINDOWS\system32\cmd.exe', shows the execution of the script. It displays the prompts 'Enter the first number:' and 'Enter the second number:', followed by the user input '6' and '8' respectively. The output shows 'The sum is 14' and 'Press any key to continue . . .', with two purple cursor icons visible at the end of the lines.

Figure 25-number count SUM

2. Script (echo off voting age count): This Windows batch script checks whether a user is eligible to vote based on their age. The command `@echo off` hides the execution commands and provides a cleaner output display. The `set /p` command is used to request the user's age and store the entered value in the variable `age`. The `if` statement compares the entered age with the value 18 using the `geq` operator, which means "greater than or equal to". If the user's age is 18 or above, the script executes the `echo` command and displays the message "Your age eligible to vote."

Otherwise, the else condition runs and displays "You are not eligible to vote yet." The pause command keeps the Command Prompt window open until the user presses a key. This script demonstrates basic input handling and decision-making using Windows batch programming [17].



The image shows two screenshots. The top screenshot is a Windows batch script editor window titled 'example 2 .bat'. It contains the following code:

```
1 @echo off
2 set /p age=Enter your age :
3 if %age% geq 18 (
4 echo Your are eligible to vote.
5 ) else (
6 echo You are not eligible to vote yet.
7 )
8 pause
```

The bottom screenshot shows two Command Prompt windows. The left window shows the output of the script when the user enters '30':

```
Enter your age :30
Your are eligible to vote.
Press any key to continue . . .
```

The right window shows the output when the user enters '5':

```
Enter your age :5
You are not eligible to vote yet.
Press any key to continue . . .
```

Figure 26-voting age count output

6. Use of Linux OS Scripts

Linux OS scripts automate system tasks, manage files, configure environments, and improve operational efficiency. They enable administrators to perform repetitive processes, monitor systems, execute commands, and maintain secure, reliable, and scalable computing environments [21].

1.Script (number count sum): This Linux OS script is written using Bash scripting language to perform the addition of two numbers. The script first displays a message asking the user to enter the first number. The read command is used to store the entered value in the variable num1. Then, it requests the second number from the user and stores it in the variable num2. The arithmetic operation is performed using the expression `total=$((num1+num2))`, which adds both numbers and saves the result in the variable total [16]. Finally, the echo command displays the calculated sum on the terminal. This script demonstrates basic Linux scripting concepts, including user input, variable assignment, arithmetic operations, and output display.

```
GNU nano 2.2.6 File: dcript.sh

echo "Eneter the first number"
read num1
echo "Enter the second number"
read num2
total=$((num1+num2))
echo "The num is $total "
```

Read 7 lines

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

```
sysadmin@localhost:~/Desktop$ nano dcript.sh
sysadmin@localhost:~/Desktop$ ls
dcript.sh new.txt new.txt
sysadmin@localhost:~/Desktop$ ls
dcript.sh new.txt new.txt
sysadmin@localhost:~/Desktop$ chmod +x script.sh
chmod: cannot access 'script.sh': No such file or directory
sysadmin@localhost:~/Desktop$
sysadmin@localhost:~/Desktop$ chmod +x dcript.sh
sysadmin@localhost:~/Desktop$ ls
dcript.sh new.txt new.txt
sysadmin@localhost:~/Desktop$ ./script.sh
bash: ./script.sh: No such file or directory
sysadmin@localhost:~/Desktop$ ./script.sh
bash: ./script.sh: No such file or directory
sysadmin@localhost:~/Desktop$ nano dcript.sh
sysadmin@localhost:~/Desktop$ ./dcript.sh
Eneter the first number
3
Enter the second number
4
The num is 7
sysadmin@localhost:~/Desktop$
```

Figure 27-Addition of two numbers

2.Script (Loop Execution Output): This Linux OS script demonstrates the use of a for loop in Bash scripting to execute a task multiple times. The script first asks the user to enter the number of times they want the loop to run. The read command stores the user input in the variable count. The for loop starts with the variable i assigned the value 1 and continues until the value of i becomes equal to the specified count [25]. During each iteration, the echo command displays the current value of i. After displaying the number, the loop automatically increases the value of i by one and repeats until the condition becomes false. This script demonstrates repetition, user input handling, and loop control in Linux OS scripting.

```
GNU nano 2.2.6 File: ./loop.sh

echo "how many time you want to run ?"
read count
for ((i=1; i<=count; i++))
do
echo "$i"
done

Read 6 lines
Get Help WriteOut Read File Prev Page Cut Text Cur Pos
Exit Justify Where Is Next Page UnCut Text To Spell
```

```
sysadmin@localhost:~/Desktop$ nano loop.sh
sysadmin@localhost:~/Desktop$ chmod +x loop.sh
sysadmin@localhost:~/Desktop$ chmod 4x loop.sh
chmod: invalid mode: '4x'
Try 'chmod --help' for more information.
sysadmin@localhost:~/Desktop$ ./loop.sh
how many time you want to run ?
7
./loop.sh: line 3: syntax error: arithmetic expression required
./loop.sh: line 3: syntax error: `((i=1, i<=count, i++))'
sysadmin@localhost:~/Desktop$ nano loop.sh
sysadmin@localhost:~/Desktop$ ./loop.sh
how many time you want to run ?
7
1
2
3
4
5
6
7
sysadmin@localhost:~/Desktop$
```

Figure 28-Loop Execution Output

7. Conclusion

This report has examined how two widely used general purpose operating systems, Windows and a Linux distribution, implement a comparable set of underlying functions through markedly different interfaces. Despite these differences in presentation, the practical work carried out for Sections 2 and 3 demonstrated that the same five categories of task, namely process management, file and directory management, network testing, user management and task scheduling, are addressed by both systems using command sets that are conceptually parallel even where their syntax differs substantially. Windows tools such as tasklist and net user favour a relatively verbose, switch based syntax rooted in the operating system's long history of backward compatibility, whereas the equivalent Linux utilities such as ps and user tend toward

shorter commands that are more frequently combined with one another through piping, reflecting the traditional Unix philosophy of small tools that each do one task well.

The scripting exercises undertaken for Sections 4 and 5 reinforced this comparison at a more technical level. Writing equivalent calculator and directory audit utilities in both Batch and Bash made clear that although the underlying programming concepts, namely variables, user input, arithmetic evaluation, conditional branching and repetition, are universal to procedural scripting, their concrete implementation differs in ways that matter in practice. Batch's reliance on the label, for example, is considerably less readable than Bash's native while loop, and Batch's delayed variable expansion inside loops is a common source of error for those new to the language. Testing the Bash scripts directly during development also demonstrated the value of validating logic empirically rather than relying solely on a visual reading of the code, since the division by zero and missing directory conditions in particular behaved exactly as intended only once actually executed against real input.

Taken as a whole, this practical work has reinforced that operating system functions are not abstract concepts confined to a textbook description, but are directly observable and controllable through the command line interfaces that both Windows and Linux provide, and that the ability to automate routine administrative tasks through simple scripting is a transferable skill that applies in comparable form across otherwise very different operating system environments.

Bibliography (IEEE Style)

- [1] A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, 10th ed. Hoboken, NJ, USA: Wiley, 2018.
- [2] W. Stallings, Operating Systems: Internals and Design Principles, 9th ed. Boston, MA, USA: Pearson, 2018.
- [3] A. S. Tanenbaum and H. Bos, Modern Operating Systems, 4th ed. Boston, MA, USA: Pearson, 2015.
- [4] Microsoft, "Troubleshoot processes by using Task Manager," Microsoft Learn, 2025. [Online]. Available: <https://learn.microsoft.com/en-us/troubleshoot/windows-server/support-tools/support-tools-task-manager>
- [5] Red Hat, "Chapter 21, System Monitoring Tools," Red Hat Enterprise Linux 7 System Administrator's Guide, Red Hat Documentation. [Online]. Available at the Red Hat customer documentation portal, System Administrator's Guide, System Monitoring Tools chapter.
- [6] Microsoft, "Windows commands," Microsoft Learn, 2025. [Online]. Available: <https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/windows-commands>
- [7] Microsoft, "PowerShell documentation," Microsoft Learn, 2025. [Online]. Available: <https://learn.microsoft.com/en-us/powershell/>
- [8] C. Ramey and B. Fox, GNU Bash Reference Manual, Edition 5.3. Boston, MA, USA: Free Software Foundation, 2025. [Online]. Available: <https://www.gnu.org/software/bash/manual/bash.html>
- J. Mueller, Windows Command Line Administration Instant Reference. Indianapolis: Wiley, 2012.
- [10] B. Shotts, The Linux Command Line, 2nd ed. San Francisco: No Starch Press, 2019.
- [11] N. Matloff, The Art of Linux System Administration. San Francisco: No Starch Press, 2017.
- [12] T. Herbert, Linux Networking Cookbook. O'Reilly Media, 2017.
- [13] A. Silberschatz, P. Galvin and G. Gagne, Operating System Concepts, 10th ed. Wiley, 2024 update.
- [14] W. Stallings, Operating Systems: Internals and Design Principles, 9th ed. Pearson, 2024.
- [15] L. Xiang et al., "Nomad: Non-exclusive memory tiering via transactional page migration," ACM Operating Systems Research, 2024.
- [16] K. Mei et al., "AIOS: LLM agent operating system architecture," arXiv Computing Systems, 2024.
- [17] E. Jones, PowerShell in Action, Manning Publications, 2024.
- [18] M. Kerrisk, The Linux Programming Interface, updated ed., No Starch Press, 2024.
- [19] D. Comer, Operating System Design, CRC Press, 2024.
- [20] S. Gener et al., "Runtime integrated memory management for heterogeneous computing," 2025.

- [21] A. Tanenbaum and H. Bos, Modern Operating Systems, 5th ed., Pearson, 2025.
- [22] J. Bonwick, "Slab allocation memory management mechanism," Oracle Technical Papers.
- [23] O. Rodeh, "Copy-on-write storage mechanisms in operating systems," ACM Transactions on Storage.
- [24] Microsoft Corporation, "Windows command reference documentation," Microsoft Learn, 2024.
- [25] Microsoft, "Network diagnostics commands for Windows," Microsoft Learn, 2024.

Appendix



FUNDAMENTALS OF QUANTUM COMPUTING (LFQ101)

Menu

COURSE INTRODUCTION

About This Course

Course Introduction

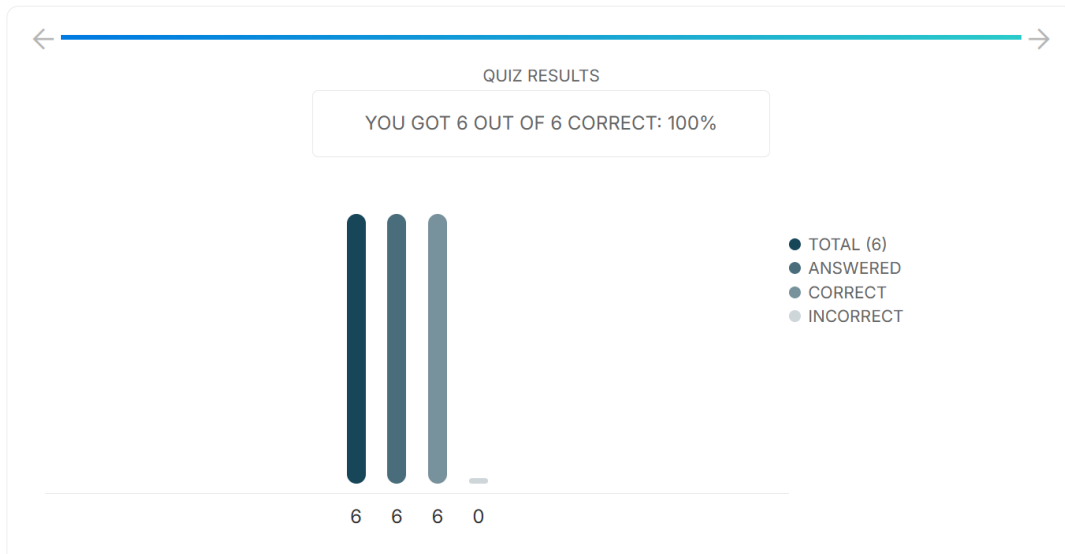


Course Introduction



Tim Serewicz

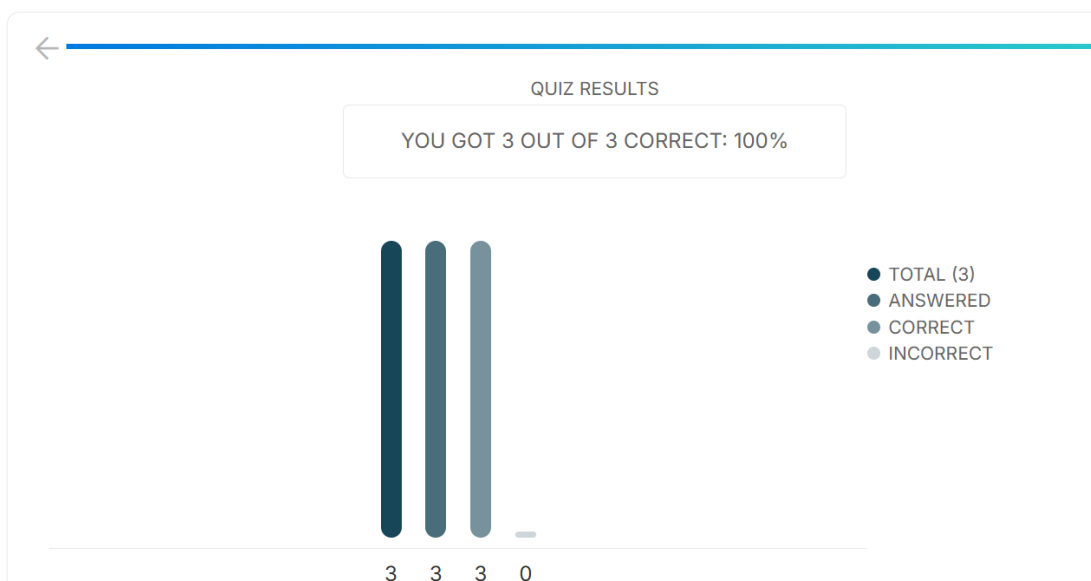
Knowledge Check



<https://trainingportal.linuxfoundation.org/learn/course/fundamentals-of-quantum-computing-lfq101/security-considerations/kno...> 📄 ★ 🔊

FUNDAMENTALS OF QUANTUM COMPUTING (LFQ101)

Knowledge Check



FUNDAMENTALS OF QUANTUM COMPUTING (LFQ101)

COURSE COMPLETION

Final Exam



TEST RESULTS

YOU GOT 10 OUT OF 10 CORRECT: 100%
YOU PASSED!



- TOTAL (10)
- ANSWERED
- CORRECT
- INCORRECT



What is Command Line Interface (CLI)?

- Command line or command-line interface is a **text-based application for viewing, handling, and manipulating files on your computer**
- Command line interface provides a way to control your computer by typing “**English-like**” **commands** instead of clicking on buttons, icons, menus etc.
- CLI much like Windows Explorer or Finder on the Mac, but without the graphical interface.
- **Other names** for the command line are: cmd, CLI, prompt, console or terminal

Linux CLI – Introduction

- The Linux command line is a text interface to your computer
- The Linux command also known as Shell, Terminal, Console, Command Prompts and many others, which is a computer program intended to interpret commands
- The Linux command allows users to execute commands by manually typing at the terminal.

Command Syntax

Commands can be run by themselves, or can accept arguments to alter their behavior. A typical syntax can look similar to this:

command [-argument] [--long-argument] file

Example:

```
John@ubuntu: ~  
John@ubuntu:~$ ls  
John_directory john_file  
John@ubuntu:~$ ls -l  
total 8  
drwxrwxr-x 2 john john 40 Oct 1 11:10 john_directory  
-rw-rw-r-- 1 john john 5120 Oct 1 11:17 john_file  
John@ubuntu:~$ ls -l -h  
total 8.0K  
drwxrwxr-x 2 john john 40 Oct 1 11:10 john_directory  
-rw-rw-r-- 1 john john 5.0K Oct 1 11:17 john_file  
John@ubuntu:~$ ls -lh john_file  
-rw-rw-r-- 1 john john 5.0K Oct 1 11:17 john_file  
John@ubuntu:~$ ls -l --human-readable john_file  
-rw-rw-r-- 1 john john 5.0K Oct 1 11:17 john_file  
John@ubuntu:~$
```



*Trawsnewid Addysg; Trawsnewid Bywydau
Transforming Education; Transforming Lives*

Basic Linux CLI Commands

file: This command will show the kind of a file passed as argument.

```
John@ubuntu: ~  
John@ubuntu:~$ file john_file.exe  
john_file.exe: ASCII text  
John@ubuntu:~$
```

In the example above, we see john_file with a ".exe" extension, the file command output the real file kind. In this case a simple text file

cd: The cd command, which stands for "Change Directory", will change your working directory to the one passed as argument

```
John@ubuntu: ~/john_directory  
John@ubuntu:~$ pwd  
/home/john  
John@ubuntu:~$ cd john_directory  
John@ubuntu:~/john_directory$ pwd  
/home/john/john_directory  
John@ubuntu:~/john_directory$
```

Note: how the prompt has changed as well.



*Trawsnewid Addysg; Trawsnewid Bywydau
Transforming Education; Transforming Lives*