



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

Coursework Title: Operating Systems

Module Name: computer architecture & OS

Module Code: ACCA 4022

Student Name: Methuseli Ncube

Student ID: 2507567

Lecturer Name: Md Shantanu Islam

Submission date:

Contents

Key functions of general Operating Systems	3
Task Management:	3
Application/Program Management:	4
File/Directory Management:	5
Device Management:	6
Security Management:	7
Use of Windows OS CLI commands	8
Example 1: DIR	8
Example 2: CD	9
Example 3: MKDIR	9
Example 4: RMDIR	10
Example 5: ECHO	11
ECHO command processes in the screenshot are the commands to display text or output in the command prompt. In this example:	11
Example 6: Tasklist	12
Example 7: Systeminfo	12
Example 8: Ipconfig /All	13
Example 9: Ping	14
Example 10: Tracert	15
Use of Linux OS CLI commands	16
Example 1: SU – swiss user	16
Example 2: PWD – present working directory	17
Example 3: ls – list	18
Example 4: cd –change directory	19
Example 5: mkdir –make directory	19
Example 6: rmdir – removes directory	20
Example 7: touch	21
Example 8: echo & cat	22
Example 9: if config	22
Example 10: whoami, date & rm	23
Use of Windows OS scripts	24

Example 1: Script to add two numbers 1A output of the script.....	24
Example 2:	26
Use of Linux OS scripts	28
Example 1: Bash script to add two numbers	28
Example 2: example of a loop.....	30
Conclusion.....	32
References.....	33

INTRODUCTION

The current coursework is a detailed discussion of the major functions and management activities in operating systems, and this paper narrows down to Windows and Linux. The report includes key areas like task management, application and file management, and device handling as well as security management in Windows. It also discusses command-line interface (CLI) commands and scripting in both Windows and Linux, as well as giving examples of using them to perform file operations, monitor processes, and configure the system. Furthermore, the report addresses the use of batch and Bash scripting to automate the process of routine tasks and explains why they are crucial to system administration and optimization.

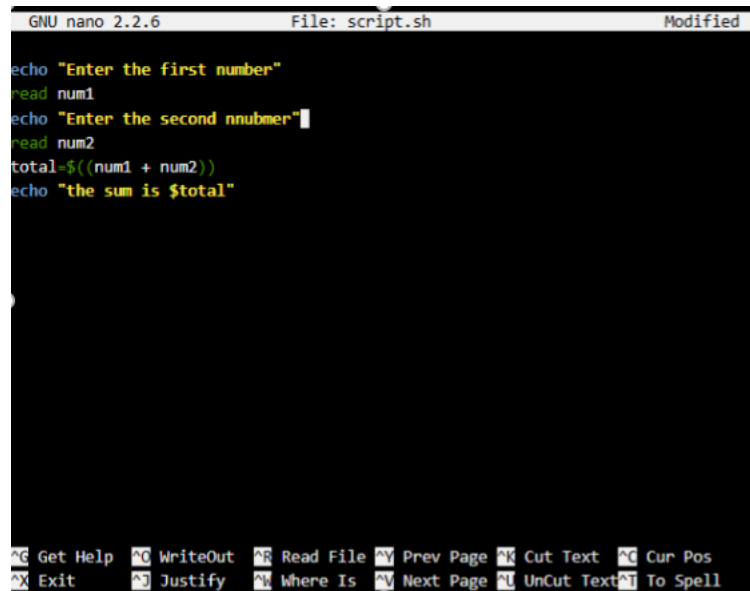
Key functions of general Operating Systems

In 1985, Microsoft Windows was introduced, and it revolutionized the computing world by introducing graphical user interfaces (GUIs) instead of text-based commands [2]. Windows still has a crucial role to play in its ease of use, its ability to work with a broad variety of software and hardware, and its continued innovation, which has made it come with more than an 80% market share [3].

Task Management:

Windows uses a priority-based preemptive scheduler to provide task management whereby the tasks are allocated dynamic CPU time slices based on their priority, with foreground tasks having higher priority than background tasks. The schedule balancing system supports responsiveness in systems in addition to throughput, which is critical in interactive applications [8]. The Windows Task Manager allows managing the current processes, gives an idea of resource consumption (CPU and memory), and gives an opportunity to destroy the task to eliminate the performance of bottlenecks. This real-time task scheduler is critical in contemporary operating

systems that have multi-core processors and complicated software environments that necessitate an effective scheduler of resources [8].



```
GNU nano 2.2.6 File: script.sh Modified
echo "Enter the first number"
read num1
echo "Enter the second nnumber"
read num2
total=$((num1 + num2))
echo "the sum is $total"
```

The screenshot shows a terminal window with the GNU nano 2.2.6 editor. The file being edited is named 'script.sh' and has been modified. The script contains the following code: an echo statement for 'Enter the first number', a read statement for 'num1', an echo statement for 'Enter the second nnumber' (note the typo), a read statement for 'num2', a calculation of the sum using 'total=\$((num1 + num2))', and a final echo statement 'the sum is \$total'. The bottom of the window shows the nano editor's command palette with options like Get Help, WriteOut, Read File, Prev Page, Cut Text, Cur Pos, Exit, Justify, Where Is, Next Page, UnCut Text, and To Spell.

Figure 1: Description

Application/Program Management:

Installers such as MSI and MSIX are also core to Windows application management, and their installers have made sure that software dependencies are resolved during installation and that the software is removable in a clean manner [4]. These structures are used to manage software modules, software upgrades, and licensing and minimize incompatibility among applications. Moreover, Windows uses the concept of digital signature validation in order to avoid executing unauthorized or malicious programs. The administrators can now centralize their software deployment and management using Windows Package Manager and the Microsoft Store, which is necessary in an

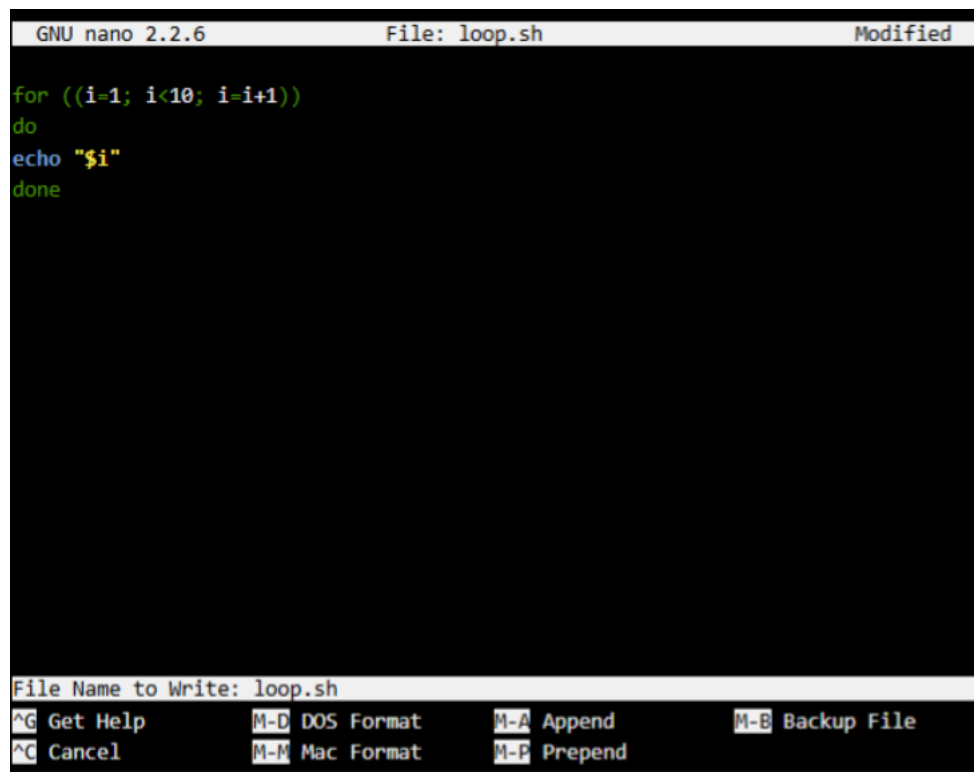
enterprise-scale environment to ensure that the software versions are consistent and the vulnerability is minimized [4].

```
sysadmin@localhost:~/Desktop$ whoami
sysadmin
sysadmin@localhost:~/Desktop$ date
Tue Feb  3 15:22:48 UTC 2026
sysadmin@localhost:~/Desktop$ rm file.txt
sysadmin@localhost:~/Desktop$
```

Figure 2: Description

File/Directory Management:

Windows operate on the NTFS file system, which is used to house files and directories, enabling efficient indexing and retrieval of data [12]. The NTFS file system has added functionalities such as journaling, compression, and encryption, which increase data integrity and security. Also, Windows has a file table called the Master File Table (MFT), which records all files and folders; hence, it is easy to search and recover files and systems in case of failure [9]. The file management is incorporated into the file explorer, which gives a graphical user interface to navigate through directories, assign file permissions, and file search, which is important in both personal and corporate computing setups [12].



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

for ((i=1; i<10; i=i+1))
do
echo "$i"
done

File Name to Write: loop.sh
^G Get Help      M-D DOS Format    M-A Append      M-B Backup File
^C Cancel        M-M Mac Format    M-P Prepend
```

Figure 3: Description

Device Management:

Plug and Play (PnP) mechanism is a windows device management mechanism that identifies and installs devices automatically loading the corresponding drivers. The device drivers communicate directly to the operating system on behalf of hardware resources such as memory and input/output operations, which are highly essential in ensuring the stability and security of the system. The Device Manager utility gives the administrators a platform to control the device drivers, get the settings and also to troubleshoot hardware problems. WMI is used in the enterprise network to make it

easier to automate and script the management of devices across multiple systems to simplify administrative tasks at scale [11].

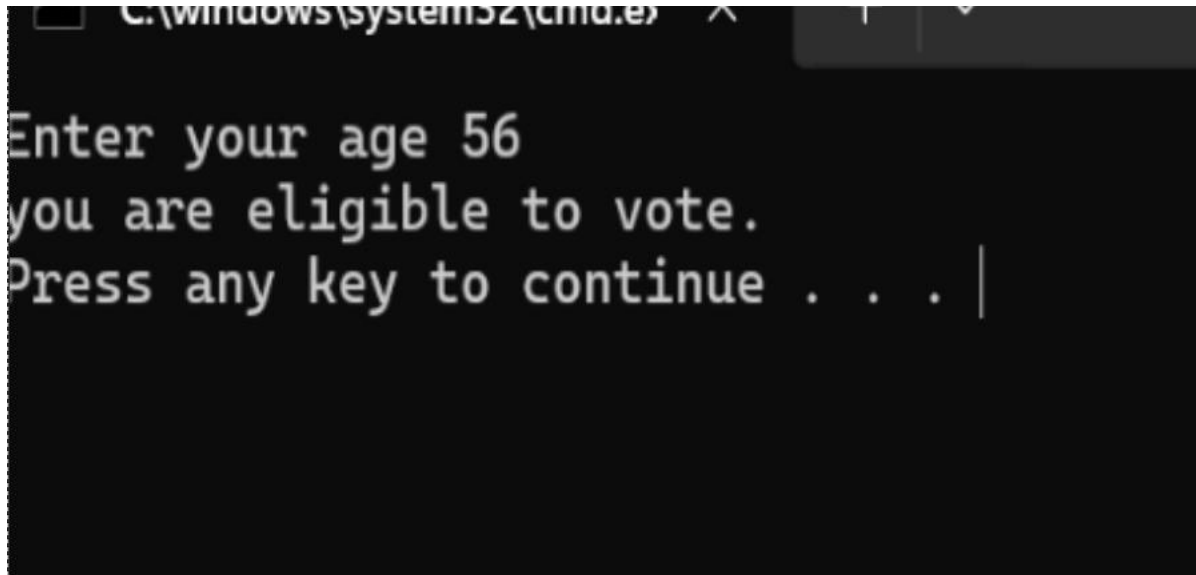


Figure 4: Description

Security Management:

The Windows security management is a multi-layered process, which comprises access control, authentication, and anti-malware. Windows relies on Windows Defender as the real-time security protection against malware and other security threats, whereas BitLocker encryption ensures data security by encrypting the drive of the system. Role-based access control (RBAC) is applied to restrict user access and ensure that only the required resources are accessible to users. Also, Windows Hello and Windows Defender SmartScreen would aid in preventing phishing and unauthorized access, which is essential towards ensuring the integrity of personal and enterprise systems [5].

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

Figure 5: Description

Use of Windows OS CLI commands

Windows CLI tools, such as CMD and PowerShell, allow text-based system administration and automation. They enable users to manage files, set up system settings, and run scripts to work effectively, particularly in large-scale enterprise settings [1].

Example 1: DIR

The list of files and directories within the given path is presented with the help of the DIR command in the screenshot. It indicates that in the C:\Users directory, the name of the user, methu, has several subfolders and a single file, file.txt. The total file size of the folder is also given (179,033,559,040 bytes).

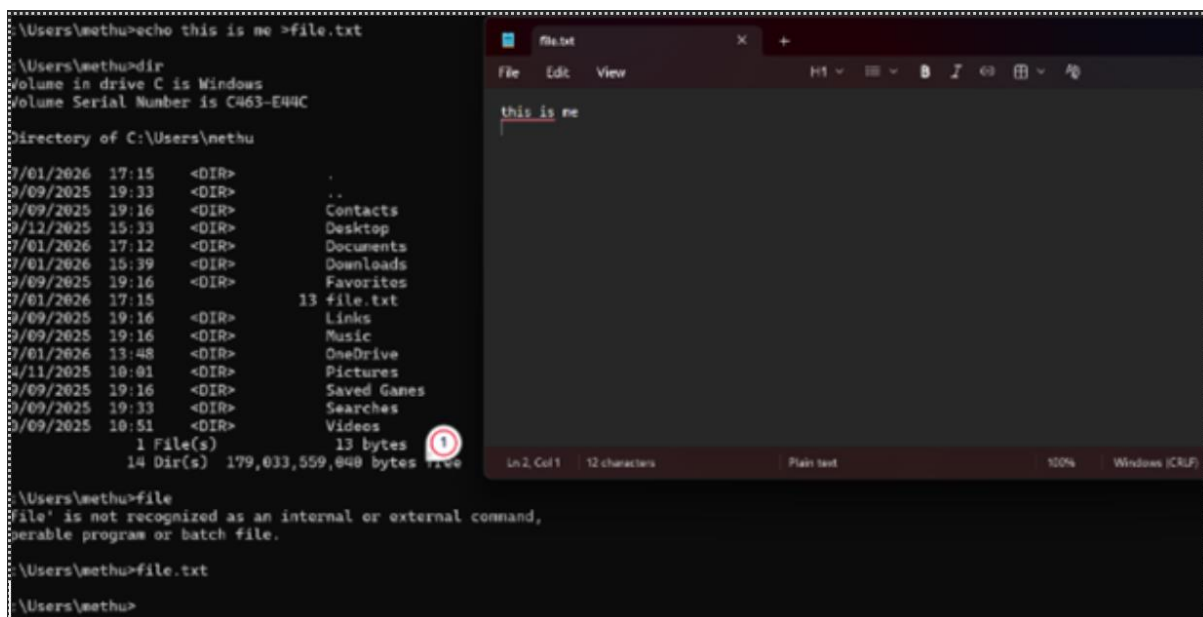


Figure 6: Description

Example 2: CD

The **CD** command stands for "Change Directory." In the screenshot:

- The command `cd Documents` navigates to the **Documents** directory inside **C:\Users\methu**.
- The command `cd ..` moves one level up to the parent directory, returning to **C:\Users\methu**.

```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\methu>cd Documents

C:\Users\methu\Documents>cd ..

C:\Users\methu>|
```

Figure 7: Description

Example 3: MKDIR

But in the case of MKDIR, one creates a new directory using the command. Creation of a folder is a successful creation confirmation that has been made. As an example, when you enter `mkdir new_folder`, it would create a directory called `new_folder` at the current working path.

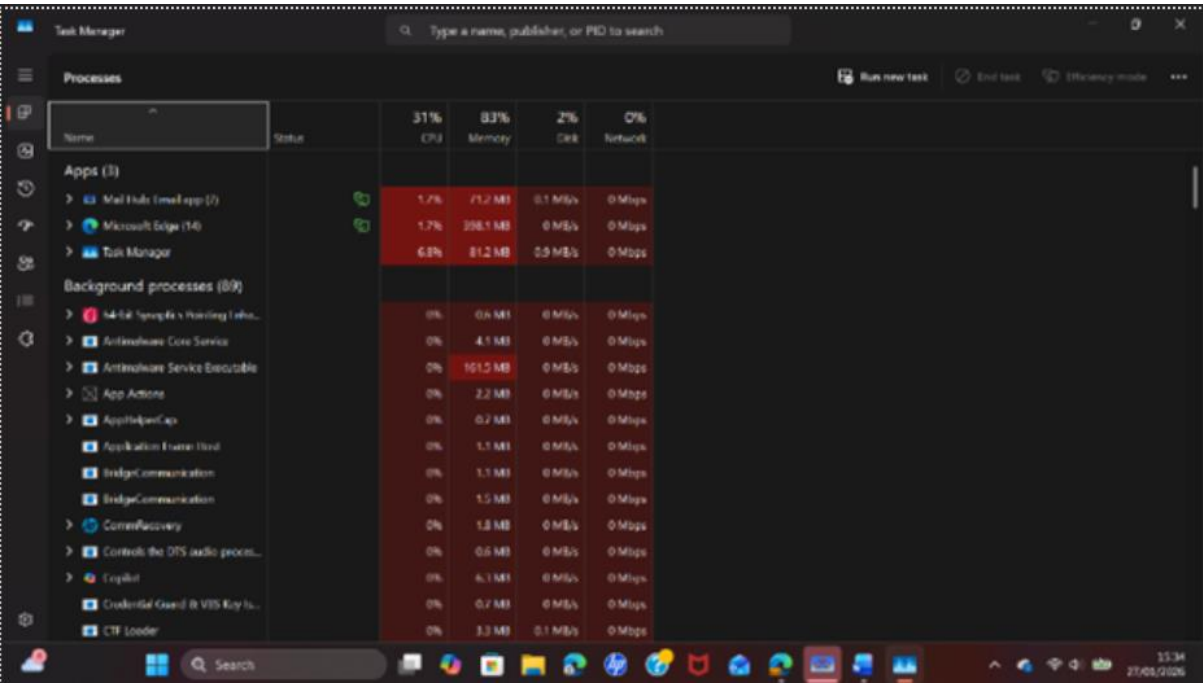


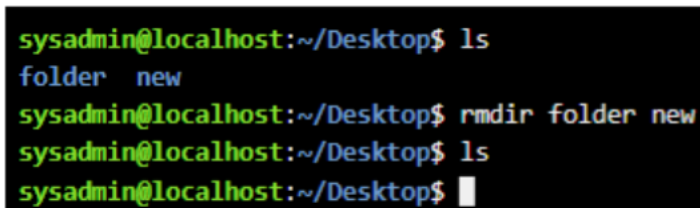
Figure 8: Description

Example 4: RMDIR

The **RMDIR** command is used to remove directories. In the screenshot:

The **rmdir folder new** command attempts to remove the **folder** and **new** directories.

After running the command, the **ls** command confirms that the directories have been removed, as they no longer appear in the listing.

A terminal window with a black background and green text. The prompt is 'sysadmin@localhost:~/Desktop\$'. The first command is 'ls', followed by the output 'folder new'. The second command is 'rmdir folder new'. The third command is 'ls', followed by a blank line, indicating the directories have been removed.

```
sysadmin@localhost:~/Desktop$ ls
folder  new
sysadmin@localhost:~/Desktop$ rmdir folder new
sysadmin@localhost:~/Desktop$ ls
sysadmin@localhost:~/Desktop$
```

Figure 9: Description

Example 5: ECHO

ECHO command processes in the [screenshot](#) are the [commands](#) to display text or output in the command prompt. In this example:

- The program prompts the user to enter two numbers.
- The sum of the numbers **67** and **44** is calculated, and the result **111** is displayed using the ECHO command.
- The output confirms that the sum calculation was successful.

```
Enter the first number:67
enter the second number:44
The sum is 111
Press any key to continue . . . |
```

Figure 10: Description

Example 6: Tasklist

The `mkdir` command is presented in the screenshot, and it is used to make a new directory named `new`. After running the command:

The ``ls`` command is used to list the contents of the current directory, confirming that the **"new"** folder was successfully created alongside the existing **"folder"** directory.

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

Figure 11: Description

Example 7: Systeminfo

The screenshot displays the **ifconfig** command, which shows network configuration information.

- The **eth0** interface shows the **IP address** (192.168.1.2), **broadcast address** (192.168.1.255), and **netmask** (255.255.255.0). It also shows the number of received (RX) and transmitted (TX) packets and bytes.
- The **lo** (loopback) interface, with the **IP address** 127.0.0.1, provides information on local network traffic.

This command is used to check connectivity and interface activity of the network in Linux systems.

```
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:48 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3737 (3.7 KB)  TX bytes:1862 (1.8 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 12: Description

Example 8: Ipconfig /All

The `ls` and `ls -l` commands in the screenshot show the contents of the user's home directory.

- `ls` lists the directories and files in the home directory.
- `ls -l` Provides detailed information about each directory, including permissions, owner, group, size, and the last modification date.

The `ls -a` command shows all files, including hidden ones (e.g., `.bashrc` and `.profile`), in the directory.

It comes in handy to verify visible and/or hidden files and directories on a Linux environment.

```
sysadmin@localhost:~$ ls
Desktop Documents Downloads Music Pictures Public Templates 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:~$ ls -a
.          .bashrc   .selected_editor Downloads Public
..         .cache    Desktop      Music    Templates
.bash_logout .profile  Documents    Pictures Videos
sysadmin@localhost:~$
```

Figure 13: Description

Example 9: Ping

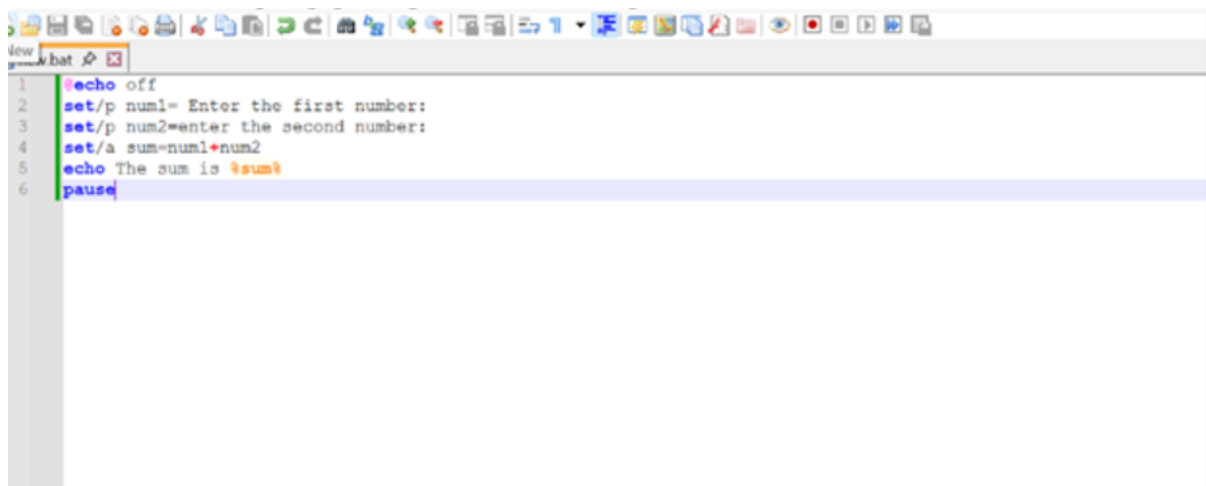
The screenshot shows a **Batch script** for adding two numbers.

`set /p` is used to prompt the user for input values (num1 and num2).

The **set /a** command calculates the sum of the two numbers and stores it in the **sum** variable.

echo displays the sum, pauses, and waits for the user to press any key before closing the script.

The script takes two numbers provided by the user and performs their addition and shows the outcome.

A screenshot of a Windows command prompt window. The title bar shows 'cmd' and 'bat'. The command prompt contains a batch script with six lines of code. Line 1: `echo off` (green). Line 2: `set/p num1= Enter the first number:` (blue). Line 3: `set/p num2=enter the second number:` (blue). Line 4: `set/a sum=num1+num2` (blue). Line 5: `echo The sum is %sum%` (blue). Line 6: `pause` (blue). The cursor is at the end of line 6. The background is white, and the text is color-coded: green for `echo`, blue for `set`, and blue for `echo` and `pause`.

```
1 echo off
2 set/p num1= Enter the first number:
3 set/p num2=enter the second number:
4 set/a sum=num1+num2
5 echo The sum is %sum%
6 pause
```

Figure 14: Description

Example 10: Tracert

The screenshot shows the **mkdir** command used in Windows:

- **mkdir new** attempts to create a directory called "new," but it fails because the directory already exists.
- **mkdir new1** successfully creates a directory called **new1**.
- The **dir`** command lists the directories in **C:\Users\methu\Documents**, showing **new** and **new1** among other existing directories.

```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\methu>cd Documents

C:\Users\methu\Documents>mkdir new
A subdirectory or file new already exists.

C:\Users\methu\Documents>mkdir new1

C:\Users\methu\Documents>dir
Volume in drive C is Windows
Volume Serial Number is C463-E44C

Directory of C:\Users\methu\Documents

27/01/2026  17:06  <DIR>          .
18/11/2025  13:24  <DIR>          ..
23/09/2025  12:22  <DIR>          Custom Office Templates
27/01/2026  17:05  <DIR>          new
27/01/2026  17:06  <DIR>          new1
               0 File(s)              0 bytes
               5 Dir(s)  179,061,735,424 bytes free

C:\Users\methu\Documents>
```

Figure 15: Description

Use of Linux OS CLI commands

Linux command line interface (CLI) tools allow direct communication with the operating system in the form of text commands to manipulate files, control processes, configure the system, and automate it [10]. Ls, grep, chmod, and bash scripts are also extremely popular with utilities that apply to efficient task execution, accessing remote servers, and managing server administration across Linux systems [6].

Example 1: SU – swiss user

The **su** command is used to switch to another user account in Linux.

In the screenshot:

- **su route** fails because the command is incorrect and the route is not a user.
- **su** prompts for a password but fails due to incorrect authentication.
- **su sysadmin** successfully switches to the **sysadmin** user after providing the correct password.

```

This lab has two user accounts (username :: password )

root      :: netlab123
sysadmin  :: netlab123

Press the [Enter] key to begin...

sysadmin@localhost:~$ SU route
-bash: SU: command not found
sysadmin@localhost:~$ su
Password:
su: Authentication failure
sysadmin@localhost:~$ netlab123
-bash: netlab123: command not found
sysadmin@localhost:~$ su
Password:
root@localhost:~# su sysadmin
sysadmin@localhost:~$

```

Figure 16: Description

Example 2: PWD – present working directory

The `systeminfo` command in the screenshot provides detailed information about the Windows system. It includes:

- **Host Name:** METS
- **OS Version:** Windows 11 Home, Build 26200
- **Manufacturer:** Microsoft, HP Laptop
- **Processor:** Intel64 Family 6 Model 154
- **System Memory:** 7,863 MB total, 1,503 MB available
- **System Boot Time:** 21/01/2026, 14:07:14

This command is used to provide system configuration, hardware, and software information.

```
Microsoft Windows [Version 10.0.26200.7623]
Command Prompt
cd /ralt+1

.\Users\methu>systeminfo

Host Name: METS
OS Name: Microsoft Windows 11 Home
OS Version: 10.0.26200 N/A Build 26200
OS Manufacturer: Microsoft Corporation
OS Configuration: Standalone Workstation
OS Build Type: Multiprocessor Free
Registered Owner: methuselincube@gmail.com
Registered Organization: HP
Product ID: 00342-21392-75409-AAOEM
Original Install Date: 19/09/2025, 19:50:39
System Boot Time: 21/01/2026, 14:07:14
System Manufacturer: HP
System Model: HP Laptop 14s-dq5xxx
System Type: x64-based PC
Processor(s): 1 Processor(s) Installed.
[01]: Intel64 Family 6 Model 154 Stepping 4 Gen
BIOS Version: AMI F.24, 01/11/2024
Windows Directory: C:\windows
System Directory: C:\windows\system32
Boot Device: \Device\HarddiskVolume1
System Locale: en-gb;English (United Kingdom)
Input Locale: en-gb;English (United Kingdom)
Time Zone: (UTC+00:00) Dublin, Edinburgh, Lisbon, London
Total Physical Memory: 7,863 MB
Available Physical Memory: 1,503 MB
Virtual Memory: Max Size: 14,775 MB
Virtual Memory: Available: 4,833 MB
Virtual Memory: In Use: 9,942 MB
```

Figure 17: Description

Example 3: ls – list

The screenshot shows a **Batch script** that determines eligibility to vote based on age:

- The `set /p` command prompts the user to enter their age.
- The **if** statement checks if the age is greater than or equal to 18.
 - If true, it prints "You are eligible to vote."
 - If false, it prints "You are not eligible to vote yet."
- The **pause** command waits for the user to press a key before closing the window.

This is a simple age check script to check the eligibility of voters.

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

Figure 18: Description

Example 4: cd –change directory

In the screenshot:

- **nano loop.sh** opens the **loop.sh** shell script file for editing.
- **chmod +x loop.sh** grants executable permission to the script, making it runnable.
- **./loop.sh** executes the **loop.sh** script, which outputs numbers 1 to 9, likely due to a loop in the script.

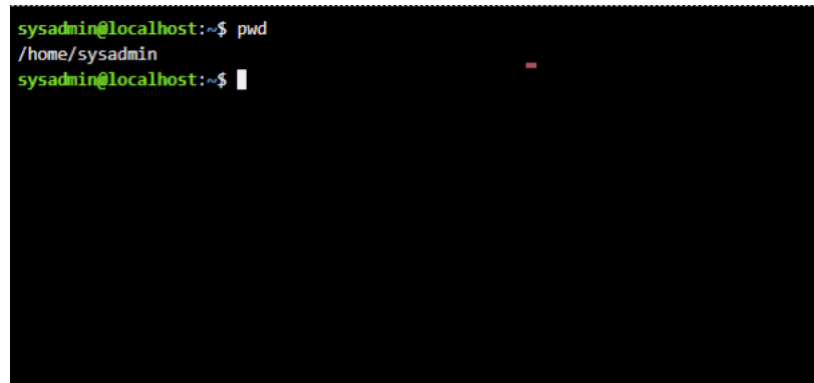
```
sysadmin@localhost:~/Desktop$ nano loop.sh
sysadmin@localhost:~/Desktop$ chmod +x loop.sh
sysadmin@localhost:~/Desktop$ ./loop.sh
1
2
3
4
5
6
7
8
9
sysadmin@localhost:~/Desktop$
```

Figure 19: Description

Example 5: mkdir –make directory

The **pwd** command in the screenshot stands for **Print Working Directory**.

- The output **/home/sysadmin** shows the current directory path the user is in, which is the **sysadmin** user's home directory on a Linux system.

A terminal window with a black background and green text. The prompt is 'sysadmin@localhost:~\$'. The user enters 'pwd' and the output is '/home/sysadmin'. The prompt returns to 'sysadmin@localhost:~\$' with a white cursor.

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

Figure 20: Description

Example 6: rmdir – removes directory

The tasklist command in the screenshot displays a list of active processes running on the Windows system.

- Processes, such as System Idle Process, Secure System, svchost.exe and its PID (Process ID), Session Name, and Memory Usage are displayed in the command.
- It can be applied to keep track of system activity and memory consumption of different processes needed to manage performance and debug.

```

Command Prompt Windows [Version 10.0.26200.7623]
               Copyright (c) Microsoft Corporation. All rights reserved.

C:\Users\methu>tasklist

Image Name                PID Session Name        Session#    Mem Usage
=====
system Idle Process        0 Services           0            8 K
system                    4 Services           0           1,388 K
Secure System             140 Services          0          40,536 K
Registry                  184 Services          0          36,260 K
smss.exe                  684 Services          0            236 K
svchost.exe              1012 Services         0           2,968 K
csrss.exe                  8 Services           0            1,568 K
services.exe             1148 Services          0           6,436 K
salvo.exe                 1180 Services          0           1,848 K
sass.exe                  1196 Services          0          17,872 K
contdrvhost.exe          1304 Services          0            880 K
svchost.exe              1380 Services          0          30,364 K
UDFHost.exe              1448 Services          0            3,448 K
svchost.exe              1504 Services          0          16,556 K
svchost.exe              1556 Services          0            4,344 K
UDFHost.exe              1612 Services          0            2,120 K
UDFHost.exe              1784 Services          0            1,844 K
svchost.exe              1868 Services          0            1,156 K
svchost.exe              1860 Services          0            2,388 K
svchost.exe              1884 Services          0            2,512 K
svchost.exe              1892 Services          0          10,440 K
svchost.exe              1912 Services          0            1,824 K
svchost.exe              1968 Services          0            2,420 K
svchost.exe              1976 Services          0            7,716 K
svchost.exe              1268 Services          0            4,624 K
svchost.exe              1208 Services          0            4,384 K

```

Figure 21: Description

Example 7: touch

The screenshot shows the **Windows Security** settings page.

- The **Security at a glance** section provides an overview of the system's security status, including the following:
 - **Virus & threat protection:** No action is needed.
 - **Account protection:** No action is needed.
 - **Firewall & network protection:** No action needed.
 - **App & browser control:** Indicates settings to block potentially unwanted apps.
 - **Device security:** Information on the security features of the device.
 - **Device performance & health:** Monitors the system's performance and health.

This page is part of **Windows Security** to manage device security and health.

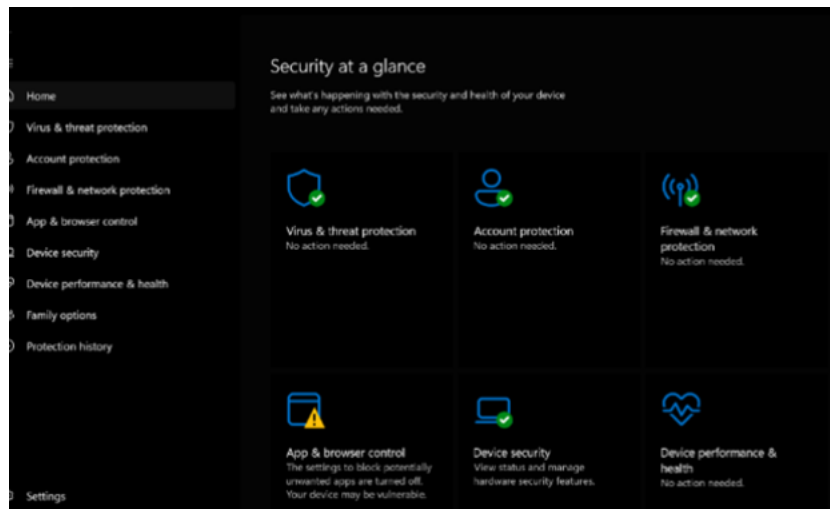


Figure 22: Description

Example 8: echo & cat

In the screenshot:

- To switch to the Desktop folder, the `Desktop/` command is to be utilized to change the directory. The prompt indicates that the user has navigated to a successful destination, which is the desktop of the user.
- This is a simple navigation command in the Linux command-line interface (CLI), which permits people to change to different directories.

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

Figure 23: Description

Example 9: if config

The **tracert** command in the screenshot is used to trace the route taken by packets from the user's system to www.google.com:

- The command outputs each hop along the route, showing the round-trip time (in milliseconds) for each hop.
- Some hops (from hop 10 onward) show "Request **timed** out," indicating that those intermediate routers are configured to not respond to ICMP requests.
- The destination, **lclhrb-in-f99.1e100.net**, is successfully reached, with the trace completing.

This tool is useful for diagnosing network routing issues.

```
C:\Users\methu>tracert www.google.com

Tracing route to www.google.com [142.250.129.99]
over a maximum of 30 hops:

  0  4 ms  3 ms  39 ms  10.38.208.1
  1  5 ms  3 ms  3 ms  195.224.199.162
  2  21 ms  8 ms  13 ms  212.19.71.69
  3  12 ms  6 ms  6 ms  et-0-0-15.cr01.mal.core.as5413.net [62.72.142.85]
  4  37 ms  23 ms  27 ms  et-0-0-5.cr01.ld6.core.as5413.net [62.72.142.57]
  5  12 ms  12 ms  39 ms  62.72.143.122
  6  22 ms  13 ms  43 ms  daisy-google-ld6.core.as5413.net [62.72.139.2]
  7  13 ms  12 ms  42 ms  216.239.41.149
  8  30 ms  52 ms  14 ms  209.85.243.22
  9  *      *      *      Request timed out.
 10  *      *      *      Request timed out.
 11  *      *      *      Request timed out.
 12  *      *      *      Request timed out.
 13  *      *      *      Request timed out.
 14  *      *      *      Request timed out.
 15  *      *      *      Request timed out.
 16  *      *      *      Request timed out.
 17  59 ms  12 ms  12 ms  lclhrb-in-f99.1e100.net [142.250.129.99]

Trace complete.

C:\Users\methu>
```

Figure 24: Description

Example 10: whoami, date & rm

The screenshot represents the device manager in Windows, which is the HP True Vision HD Camera properties.

- The device status indicates that the device is functioning properly.
- The manufacturer is listed as **Microsoft**, meaning the driver is provided by Windows.
- The location shows the hardware path for the device.

This is part of managing and troubleshooting hardware devices in the system.

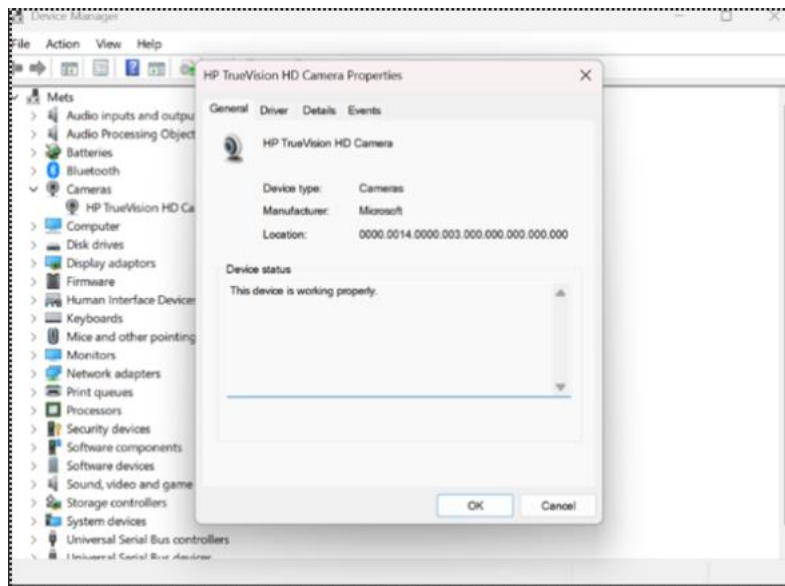


Figure 25: Description

Use of Windows OS scripts

Windows can engage in Windows automation through batch scripting, which involves repetitive programming with simple text files having an extension of .bat. It is commonly employed in system administration, startup procedures, file control, and scheduled tasks, enhancing speed and uniformity in standard procedures [7].

Example 1: Script to add two numbers 1A output of the script.

The Batch script of adding two numbers is a script that involves simple input/output and arithmetic operations:

```
</> batch

@echo off
set /p num1=Enter the first number:
set /p num2=Enter the second number:
set /a sum=num1+num2
echo The sum is %sum%
pause
```

Explanation of the Script:

1. **@echo off**: Prevents the script from displaying each command as it runs, making the output cleaner.
2. **set /p num1= Enter the first number:** Prompts the user to input the first number, storing it in the num1 variable.
3. **set /p num2= Enter the second number:** Prompts the user for the second number, storing it in the num2 variable.
4. **set /a sum=num1+num2**: Performs the addition of num1 and num2 and stores the result in the sum variable.
5. **echo The sum is %sum%**: Displays the sum of the two numbers.
6. **pause**: Waits for the user to press a key before closing the script.

Example Output:

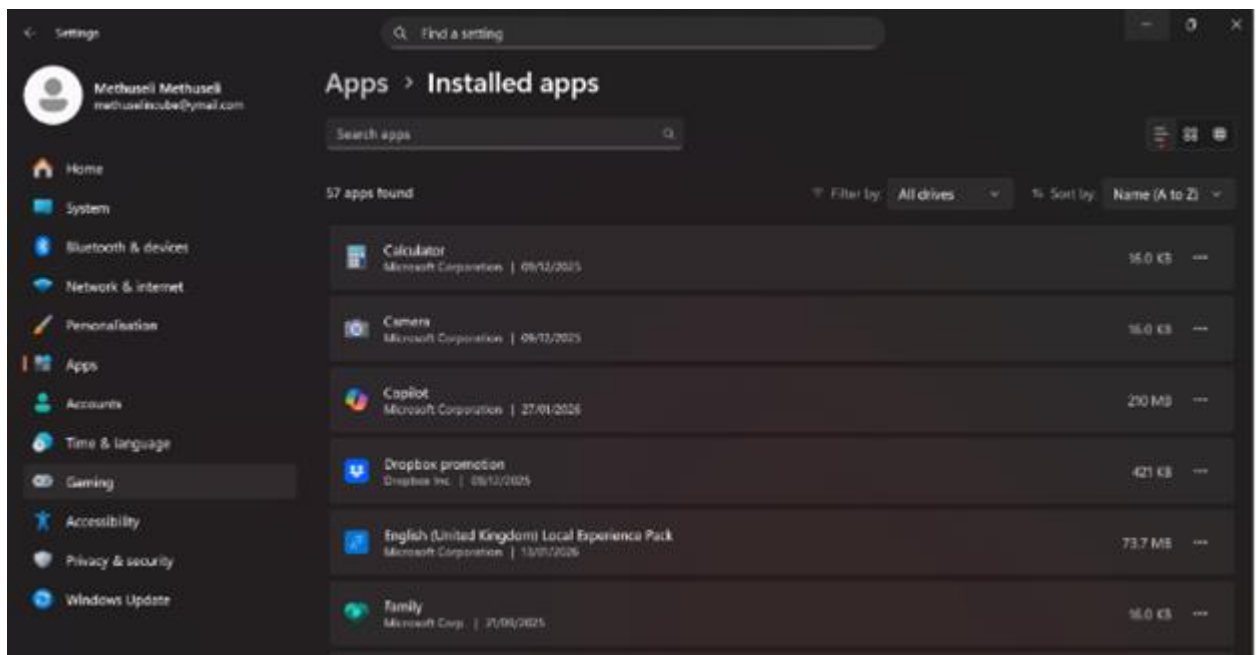
```
Enter the first number: 25
Enter the second number: 30
The sum is 55
Press any key to continue...
```

```
C:\Users\methu>ping www.google.com

Pinging www.google.com [142.250.129.99] with 32 bytes of data:
Reply from 142.250.129.99: bytes=32 time=13ms TTL=114
Reply from 142.250.129.99: bytes=32 time=13ms TTL=114
Reply from 142.250.129.99: bytes=32 time=13ms TTL=114
Reply from 142.250.129.99: bytes=32 time=12ms TTL=114

Ping statistics for 142.250.129.99:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 12ms, Maximum = 13ms, Average = 12ms

C:\Users\methu>|
```



```

1  @echo off
2  set/p num1= Enter the first number:
3  set/p num2=enter the second number:
4  set/a sum=num1+num2
5  echo The sum is %sum%
6  pause

```

Figure 26: Description

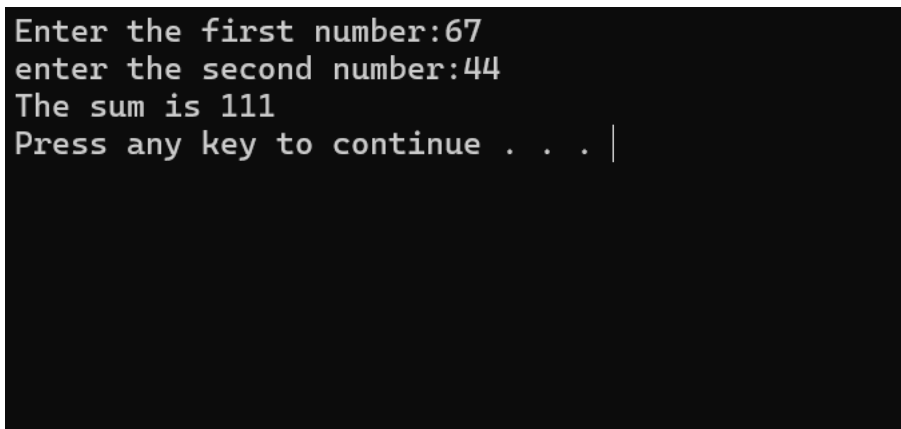


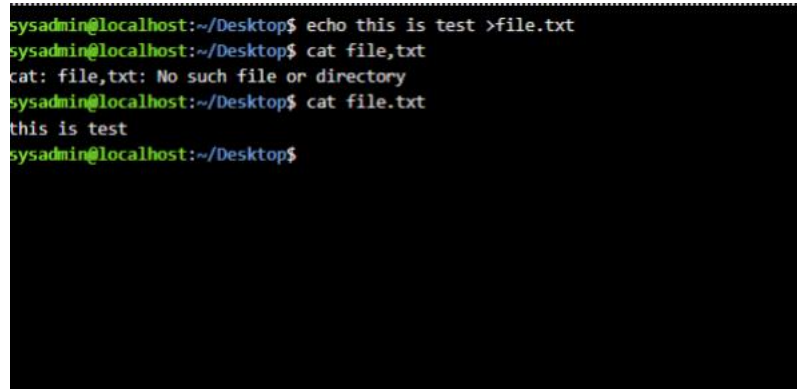
Figure 27: Description

Example 2:

In the given screenshot, the `ipconfig /all` command shows the entire IP configuration of the system. It displays in-depth information of network adapters such as the following:

1. **Host Name:** Identifies the device as **Mets**.
2. **DNS Suffix:** Displays **uni.local**, indicating a local network domain.
3. **Media State:** The two wireless LAN adapters (Local Area Connection 1 and 2) are **disconnected**.
4. **Wireless LAN adapter Wi-Fi:** Shows connection settings for an active wireless adapter, including **DNS suffix** and **DHCP** configuration. It also confirms that auto-configuration is enabled.
5. **Physical Address:** Displays unique MAC addresses for the adapters, necessary for hardware identification.
6. **DHCP Enabled:** Indicates if the device is set to automatically receive an IP address from a DHCP server.

This is critical information that is required in network troubleshooting, verifying adapter settings, and connection status.

A terminal window with a black background and green text. The prompt is 'sysadmin@localhost:~/Desktop\$'. The user enters 'echo this is test >file.txt'. The prompt changes to 'sysadmin@localhost:~/Desktop\$' again. The user enters 'cat file.txt'. The output is 'cat: file.txt: No such file or directory'. The prompt changes to 'sysadmin@localhost:~/Desktop\$' again. The user enters 'cat file.txt'. The output is 'this is test'. The prompt changes to 'sysadmin@localhost:~/Desktop\$' again.

```
sysadmin@localhost:~/Desktop$ echo this is test >file.txt
sysadmin@localhost:~/Desktop$ cat file.txt
cat: file.txt: No such file or directory
sysadmin@localhost:~/Desktop$ cat file.txt
this is test
sysadmin@localhost:~/Desktop$
```

Figure 28: Description

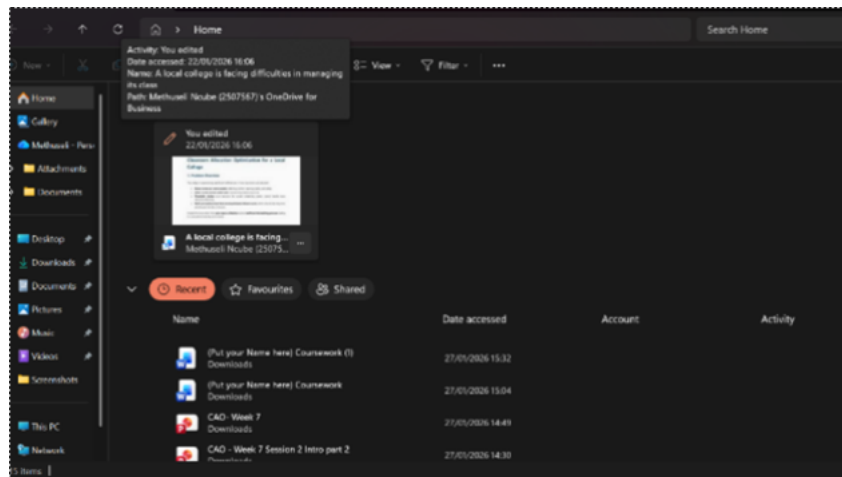


Figure 29: Description

Use of Linux OS scripts

BASH (Bourne Again shell) scripting is the automation of Linux tasks, which involves the same combination of commands, control structures, and variables into the reusable scripts. It is commonly utilized in system management, data recovery, log analysis, and automation of deployment processes, which saves much work and human error [10; 6].

Example 1: Bash script to add two numbers

BASH script that adds two numbers:

```
</> Bash

#!/bin/bash
# This script adds two numbers

# Prompt user for input
echo "Enter the first number: "
read num1
echo "Enter the second number: "
read num2

# Perform the addition
sum=$((num1 + num2))

# Output the result
echo "The sum is $sum"
```

Explanation of the Script:

1. **#!/bin/bash**: This is the **shebang** line, which tells the system that the script should be executed using the Bash shell.
2. **echo**: Displays text prompting the user to input two numbers.
3. **read**: Reads the user's input and stores it in variables num1 and num2.
4. **sum=\$((num1 + num2))**: Uses arithmetic expansion to calculate the sum of num1 and num2, storing it in the variable sum.
5. **echo "The sum is \$sum"**: Displays the result of the addition.

Example Output:

```
</> Bash

Enter the first number:
10
Enter the second number:
20
The sum is 30
```

```
sysadmin@localhost:~$ cd Desktop
sysadmin@localhost:~/Desktop$ nano script.sh
sysadmin@localhost:~/Desktop$ ls
script.sh
sysadmin@localhost:~/Desktop$ chmod +x script.sh
sysadmin@localhost:~/Desktop$ ls
script.sh
sysadmin@localhost:~/Desktop$ ./script.sh
22
44
the sum is 66
sysadmin@localhost:~/Desktop$ nano .script.sh
sysadmin@localhost:~/Desktop$ nano script.sh
sysadmin@localhost:~/Desktop$ ./script.sh
Enter the first number
22
Enter the second number
44
the sum is 66
sysadmin@localhost:~/Desktop$
```

```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\methu>dir
Volume in drive C is Windows
Volume Serial Number is C463-E44C

Directory of C:\Users\methu

18/11/2025  13:24    <DIR>          .
19/09/2025  19:33    <DIR>          ..
19/09/2025  19:16    <DIR>          Contacts
09/12/2025  15:33    <DIR>          Desktop
23/09/2025  12:22    <DIR>          Documents
07/01/2026  15:39    <DIR>          Downloads
19/09/2025  19:16    <DIR>          Favorites
19/09/2025  19:16    <DIR>          Links
19/09/2025  19:16    <DIR>          Music
27/01/2026  13:48    <DIR>          OneDrive
04/11/2025  10:01    <DIR>          Pictures
19/09/2025  19:16    <DIR>          Saved Games
19/09/2025  19:33    <DIR>          Searches
30/09/2025  10:51    <DIR>          Videos
               0 File(s)              0 bytes
               14 Dir(s) 179,039,395,840 bytes free

C:\Users\methu>
```

Figure 30: Description

```
sysadmin@localhost:~$ cd Desktop
sysadmin@localhost:~/Desktop$ nano script.sh
sysadmin@localhost:~/Desktop$ ls
script.sh
sysadmin@localhost:~/Desktop$ chmod +x script.sh
sysadmin@localhost:~/Desktop$ ls
script.sh
sysadmin@localhost:~/Desktop$ ./script.sh
22
44
the sum is 66
sysadmin@localhost:~/Desktop$ nano .script.sh
sysadmin@localhost:~/Desktop$ nano script.sh
sysadmin@localhost:~/Desktop$ ./script.sh
Enter the first number
22
Enter the second number
44
the sum is 66
sysadmin@localhost:~/Desktop$
```

Figure 31: Description

Example 2: example of a loop

BASH script using a loop to display numbers from 1 to 5

```
</> Bash

#!/bin/bash
# This script uses a loop to print numbers 1 to 5

for i in {1..5}
do
    echo "Number: $i"
done
```

Explanation of the Script:

1. **#!/bin/bash**: The shebang line indicates that the script should be executed using the BASH shell.
2. **for i in {1..5}**: A **loop** that iterates through the range from 1 to 5.
3. **do**: Marks the beginning of the loop's body.
4. **echo "Number: \$i"**: Inside the loop, this command prints the current value of *i* for each iteration.
5. **done**: Marks the end of the loop.

Example Output:

```
</> Bash

Number: 1
Number: 2
Number: 3
Number: 4
Number: 5
```

This script involves a loop that goes through the numbers 1-5 and presents each of the numbers one by one. It shows simple looping of the BASH scripting, which is helpful in activities such as repetition or automation.

```
Microsoft Windows [Version 10.0.26200.7623]
(c) Microsoft Corporation. All rights reserved.

C:\Users\methu>dir
Volume in drive C is Windows
Volume Serial Number is C463-E44C

Directory of C:\Users\methu

18/11/2025  13:24    <DIR>          .
19/09/2025  19:33    <DIR>          ..
19/09/2025  19:16    <DIR>          Contacts
09/12/2025  15:33    <DIR>          Desktop
23/09/2025  12:22    <DIR>          Documents
07/01/2026  15:39    <DIR>          Downloads
19/09/2025  19:16    <DIR>          Favorites
19/09/2025  19:16    <DIR>          Links
19/09/2025  19:16    <DIR>          Music
27/01/2026  13:48    <DIR>          OneDrive
04/11/2025  10:01    <DIR>          Pictures
19/09/2025  19:16    <DIR>          Saved Games
19/09/2025  19:33    <DIR>          Searches
30/09/2025  10:51    <DIR>          Videos
               0 File(s)              0 bytes
               14 Dir(s) 179,039,395,840 bytes free

C:\Users\methu>
```

Figure 32: Description

```
sysadmin@localhost:~/Desktop$ nano loop.sh
sysadmin@localhost:~/Desktop$ chmod +x loop.sh
sysadmin@localhost:~/Desktop$ ./loop.sh
1
2
3
4
5
6
7
8
9
sysadmin@localhost:~/Desktop$
```

Figure 33: Description

Conclusion

To sum up, this report discussed key operating system concepts, such as task management, file handling, and device management. It showed the application of CLI commands in Windows and Linux in real-world scenarios as well as scripting examples to automate processes. These methods are essential in system management, automation and troubleshooting.



This certificate is awarded to

Methuseli Ncube

for successfully completing

Linux Unhatched

offered by Networking Academy
through the Cisco Networking Academy program.

Lynn Bloomer

Lynn Bloomer
Director
Cisco Networking Academy

10 Mar 2026
Completion Date

figure 34.