**Day 6 (31 May 2025)**

**112552596 - vsisukhw - Sukhwinder Singh**

**Task 1**

**Regular expression (regex)**

Regular expressions (regex) in Linux are like patterns used for searching, matching, and manipulating text. They are commonly used with tools like grep, sed, awk, find, etc.

### **🔹 Basic Regular Expression Syntax (Linux)**

### **. (dot)** Matches any single character except a newline. Example: a.c matches abc, acc, a-c

### **\* (asterisk)** Matches 0 or more occurrences of the previous character. Example: lo\* matches l, lo, loo, looo

### **+ (plus)** Matches 1 or more occurrences of the previous character. Example: lo+ matches lo, loo, but not just l

### **? (question mark)** Matches 0 or 1 occurrence of the previous character. Example: lo? matches l or lo

### **[] (square brackets)** Matches any one character inside the brackets. Example: [abc] matches a, b, or c

### **[^] (caret inside brackets)** Matches any one character NOT inside the brackets. Example: [^0-9] matches any character except digits

### **^ (caret)** Matches the start of a line. Example: ^abc matches lines that start with abc

### **$ (dollar sign)** Matches the end of a line. Example: abc$ matches lines that end with abc

### **| (pipe)** Acts as a logical OR. Example: foo|bar matches foo or bar

### **() (parentheses)** Used to group expressions or capture groups. Example: (ab)+ matches ab, abab, ababab

### **{n,m} (curly braces)** Matches at least n and at most m occurrences of the previous character. Example: a{2,4} matches aa, aaa, or aaaa

### 

**Task 2**

**Linux OS and its Features**

### 

### **Main Features of Linux**

1. **Open Source**
   * The source code of Linux is freely available.
   * Anyone can view, modify, and distribute it.
   * Popular distributions: Ubuntu, Fedora, Debian, CentOS, Arch, etc.
2. **Free to Use**
   * Most Linux distributions are completely free.
   * No licensing fees (unlike Windows).
3. **Multi-user Capability**
   * Multiple users can log in and use the system at the same time.
   * Each user has their own permissions and environment.
4. **Multitasking**
   * Can run multiple tasks/processes at once without slowing down the system.
5. **Security**
   * File permission structure (read, write, execute).
   * Regular updates, open-source transparency, and lower virus threats.
   * Tools like iptables, SELinux, and AppArmor for added protection.
6. **Portability**
   * Linux can run on almost any hardware (phones, servers, supercomputers, Raspberry Pi).
7. **Stability and Reliability**
   * Linux systems can run for years without rebooting.
   * Ideal for servers that need high uptime.
8. **Command-Line Interface (CLI)**
   * Powerful terminal access for advanced control and automation.
   * Bash, Zsh, Fish are common shells.
9. **Modular Architecture**
   * Linux follows a monolithic kernel but can load/unload components like device drivers dynamically.
10. **Support for Programming**

* Supports almost all major programming languages (C, Python, Java, Shell, etc.).
* Built-in compilers and scripting tools.

1. **Package Management**

* Software is installed and managed via package managers like apt, yum, dnf, pacman.

1. **Community Support**

* Large and active community support through forums, GitHub, Stack Overflow, etc.

1. **File System Support**

* Supports multiple file systems like ext4, XFS, Btrfs, NTFS, FAT32, etc.

1. **Networking Capabilities**

* Linux is built for network tasks: servers, routers, firewalls, VPNs, etc.

1. **Customization**

* Highly customizable (desktops, scripts, services).
* we can even build our own version of Linux using tools like Linux From Scratch (LFS).

**Task 3**

**Kernel**

### 

The kernel is the core part of any operating system, including Linux. Its like bridge between computer's hardware and software.

It controls how everything works inside computer—but we can’t see it directly.

### In Simple Words:

Imagine computer is a factory:

* The hardware (CPU, memory, hard drive) are the machines.
* The applications (browser, editor, games) are the workers.
* The kernel is like the manager that makes sure workers can use the machines properly—without fighting, crashing, or making a mess.

**Task 4**

**Bash Full Form and Explanation**

### 

BASH stands for Bourne Again SHell

It is a command-line shell and scripting language used in most Linux distributions.

Bash is the default shell in most Linux systems. A shell is a program that takes commands from our keyboard and gives them to the operating system to execute.

**Task 5**

**Difference b/w Linux and Windows**

Linux is a free, open-source operating system that is secure, fast, and highly customizable. It’s mostly used by developers, tech users, and for servers. Windows, on the other hand, is a paid, closed-source system developed by Microsoft. It’s user-friendly, widely used in homes and offices, and supports most commercial software and games.

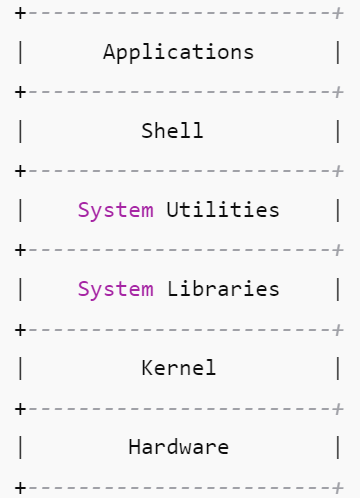
Linux is more secure and efficient, while Windows is easier for beginners and better for general use.

**Task 6**

**Basic Components of Linux**

**Basic Components of Linux (Brief)**

1. **Kernel** The core part of Linux that manages hardware, memory, and processes.
2. **System Libraries** Help programs communicate with the kernel by providing reusable functions.
3. **System Utilities** Basic commands and tools to perform common tasks like file management.
4. **Shell** The interface where users type commands; it passes these commands to the kernel.
5. **Hardware** Physical devices like CPU, memory, and storage that Linux controls through the kernel.
6. **Applications** Programs that users run, such as browsers and editors, which use the shell and libraries.



**Task 7**

**Is it legal to edit Kernal ? When we have to edit in case?**

Yes, it is completely legal to edit the Linux kernel!

Linux is open-source software released under the GNU General Public License (GPL). This license allows anyone to view, modify, and redistribute the kernel’s source code freely, as long as they follow the GPL terms (like sharing modifications under the same license).

### When would we want to edit the kernel?

We might want to edit or customize the Linux kernel in cases like:

* Adding support for new hardware that isn’t recognized by the current kernel.
* Improving performance or fixing bugs specific to our hardware or use case.
* Removing unneeded features to create a smaller, faster, or more secure kernel.
* Developing new kernel features or modules for specialized tasks.
* Experimenting or learning how an OS works at the core level.

**Task 8**

**What is LILO?**

**LILO** stands for **Linux Loader**.

It is a **boot loader** for Linux — a small program that **loads the Linux operating system** into memory when the computer starts.

**Task 9**

**What is Shell, how many shells are there and what are they?**

A **shell** is a program that acts as an interface between the user and the operating system (kernel).

It lets us to enter commands that the system can understand and execute.

### 

### **Types of Shells in Linux:**

There are **many types** of shells.

#### **1. Bash (Bourne Again SHell)**

* Most common and default on many Linux systems
* Easy to use, supports scripting
* Command: /bin/bash

#### **2. Sh (Bourne Shell)**

* The original Unix shell
* Simpler and older than Bash
* Command: /bin/sh

#### **3. Ksh (Korn Shell)**

* Combines features of Bourne and C shells
* Good for scripting and programming
* Command: /bin/ksh

#### **4. Csh (C Shell)**

* Syntax similar to C programming
* Has features like aliases and command history
* Command: /bin/csh

#### **5. Tcsh**

* An enhanced version of C Shell
* Includes improvements like auto-completion
* Command: /bin/tcsh

#### **6. Zsh (Z Shell)**

* Powerful and customizable
* Has features of Bash, Ksh, and Tcsh
* Popular for its themes and plugins (e.g., Oh My Zsh)
* Command: /bin/zsh

**Task 10**

**What is Swap Space?**

Swap space is a special area on the hard disk that the operating system uses as virtual memory when RAM is full.

In simple words,

We can imagine that RAM is our desk. When it's full of books (data), we move some to a nearby shelf (swap space) to make room.

**Task 11**

**What is Mount, how do we mount and unmount file system in Linux?**

In Linux, mounting means making a storage device (like USB, hard drive, CD, etc.) accessible to the system by attaching it to a directory (called a mount point).

Once mounted, we can read or write files from that device as if it’s part of our file system.

**For mounting:**

sudo mount /dev/sdX1 /mnt

**For un-mounting:**

sudo umount /mnt

**Task 12**

**What is chmod command and how to use it?**

**chmod** stands for "change mode".

It is used to change the permissions (read, write, execute) of a file or directory in Linux.

Linux systems control who can:

* **Read** a file (r)
* **Write** or modify a file (w)
* **Execute** a file or script (x)

chmod lets we set or modify these permissions for:

* **User (u)** – the file owner
* **Group (g)** – users in the file’s group
* **Others (o)** – everyone else

How to Use chmod:

Each permission has a number:

* Read = 4
* Write = 2
* Execute = 1

They are added together:

* Read + Write + Execute = 7
* Read + Execute = 5, etc.

Format: chmod XYZ filename

Where:

* X = User permission
* Y = Group permission
* Z = Others permission

Example:

1. chmod 755 filename

Here,

User: 7 (r+w+x), Group: 5 (r+x), Others: 5 (r+x)

1. chmod 644 myfile.txt

Here,

User: read/write,

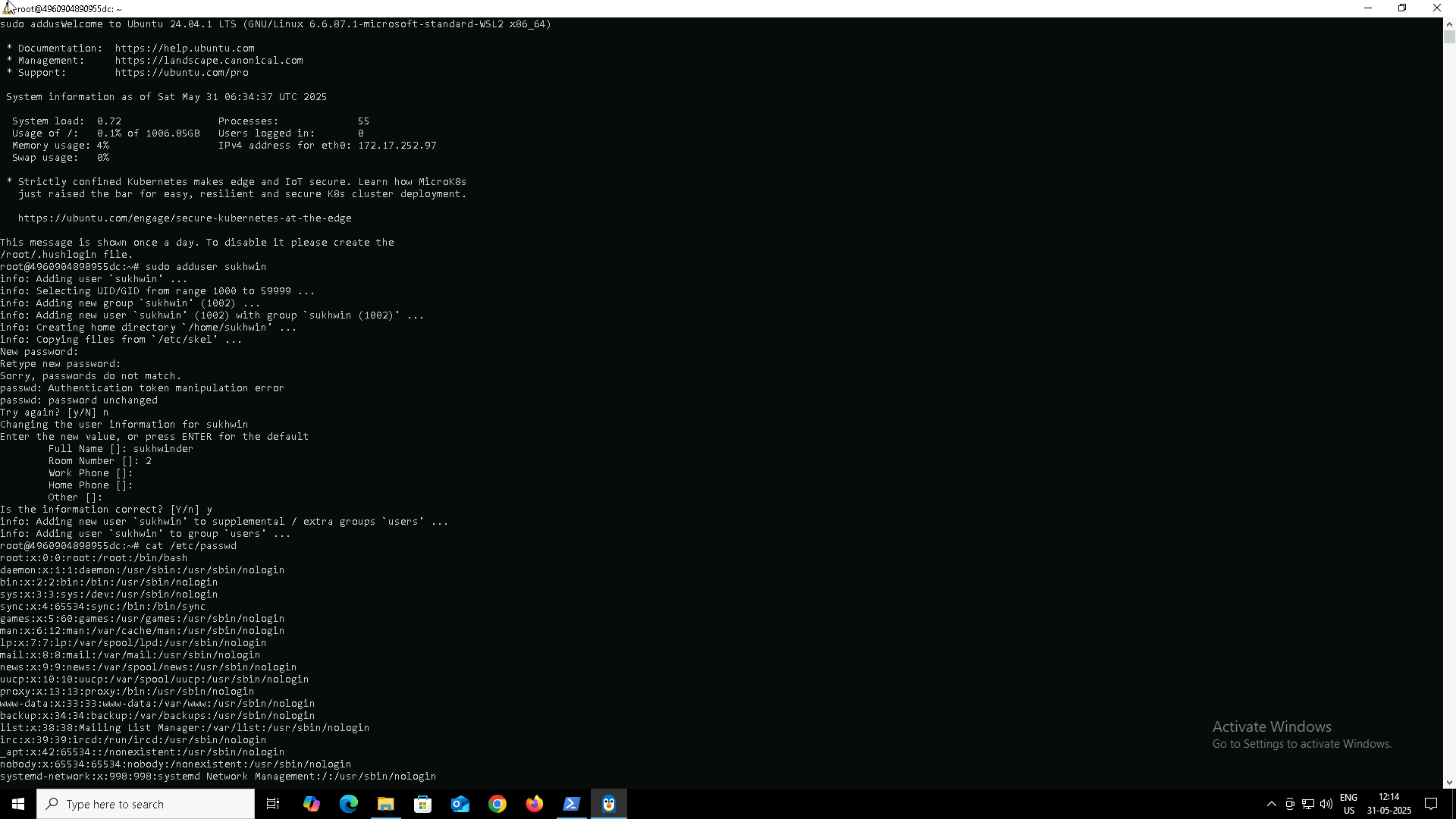
Group: read,

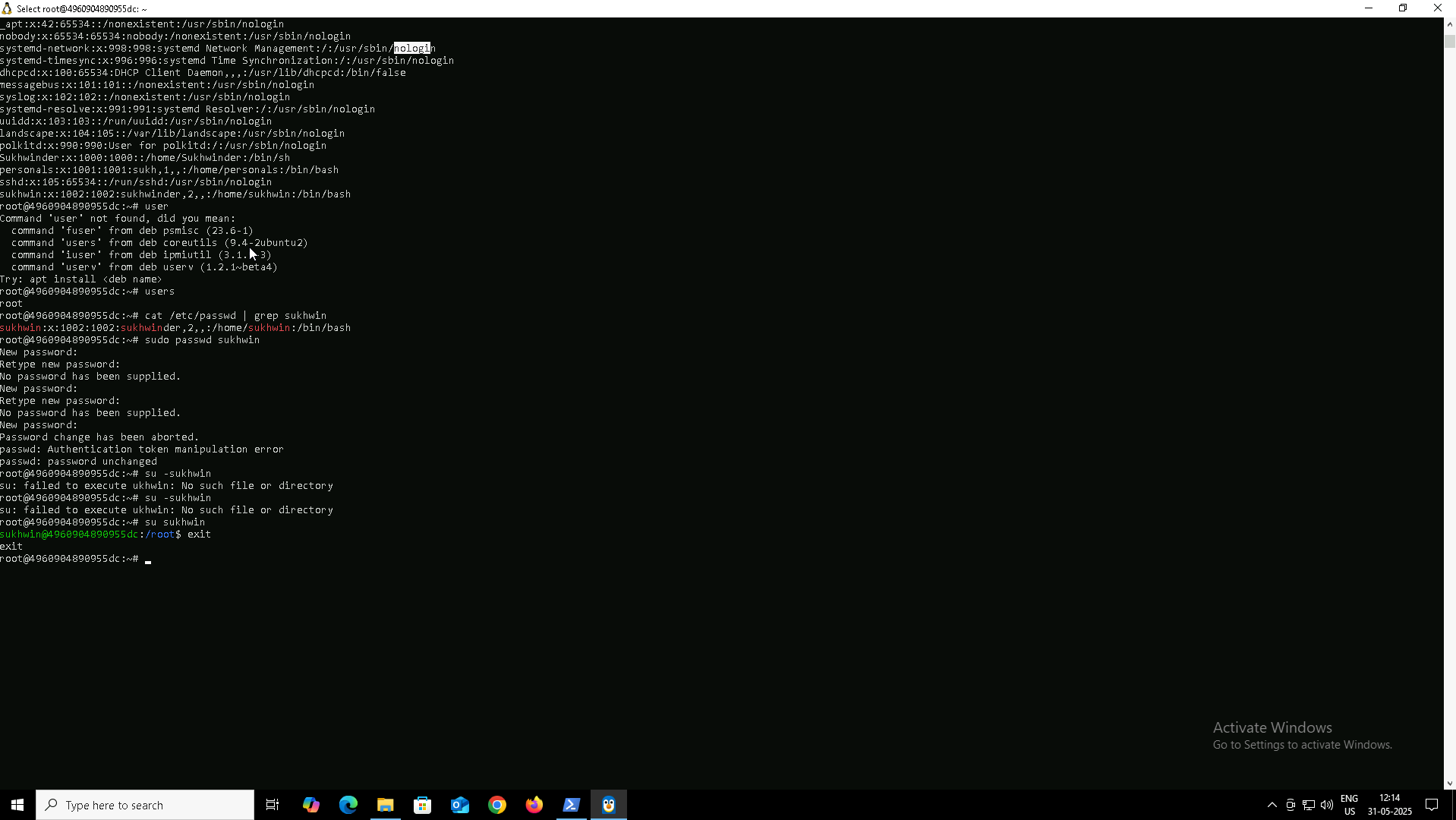
Others: read

**Task 13**

**Can we add new user , how to add it ?**

Yes , we can add new user, following are the screenshot:

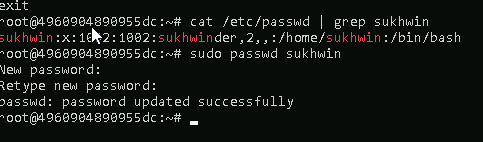




**Task 14**

**Can we change password of user ,how to do it?**

Yes , we can change password of user, following are the screenshot:



**Task 15**

**Differnce b/w Process and Thread**

**Process:**

* Independent program running with its own memory.
* Heavyweight, slower to create.
* Crashes don’t affect other processes.

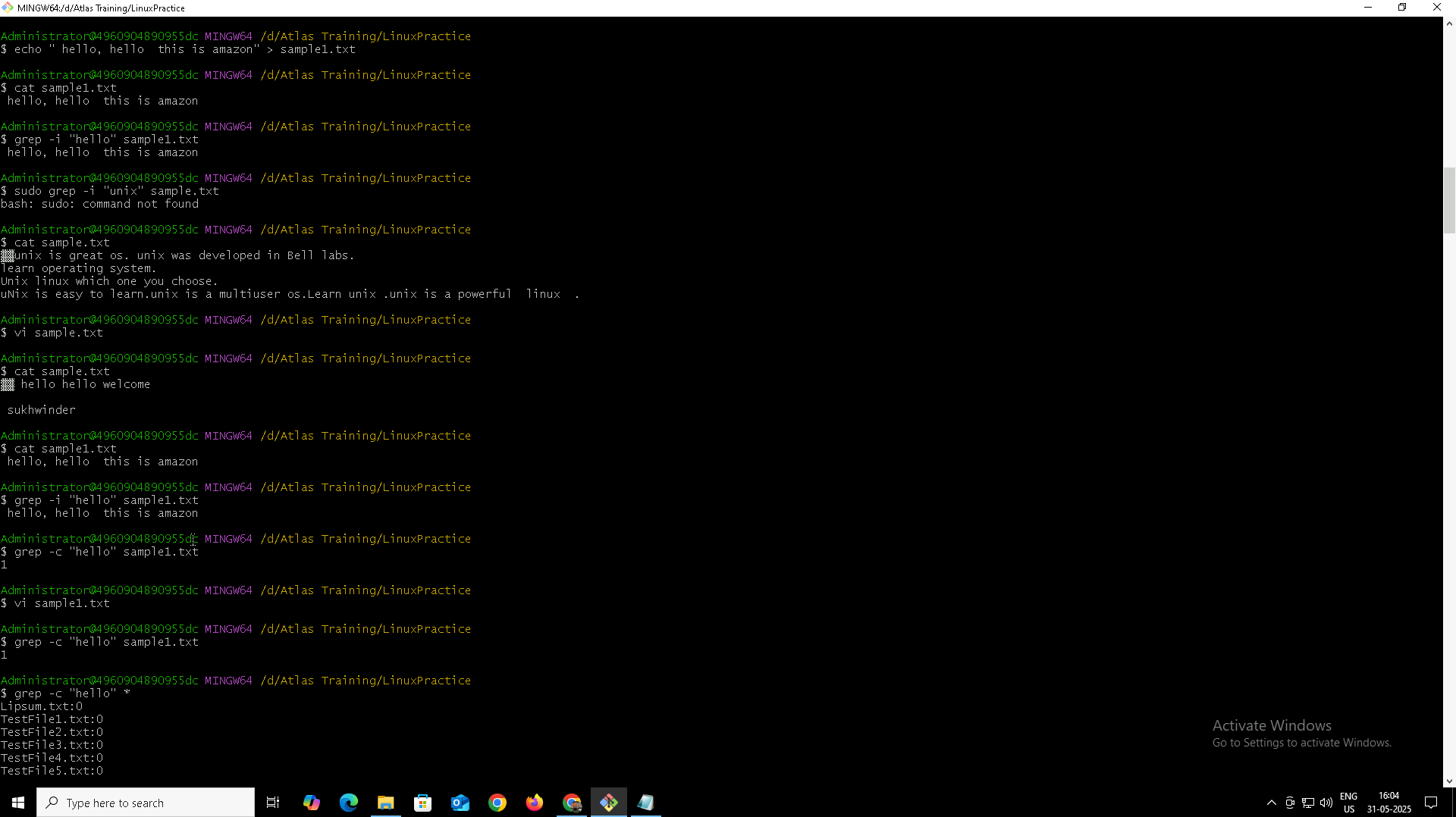
**Thread:**

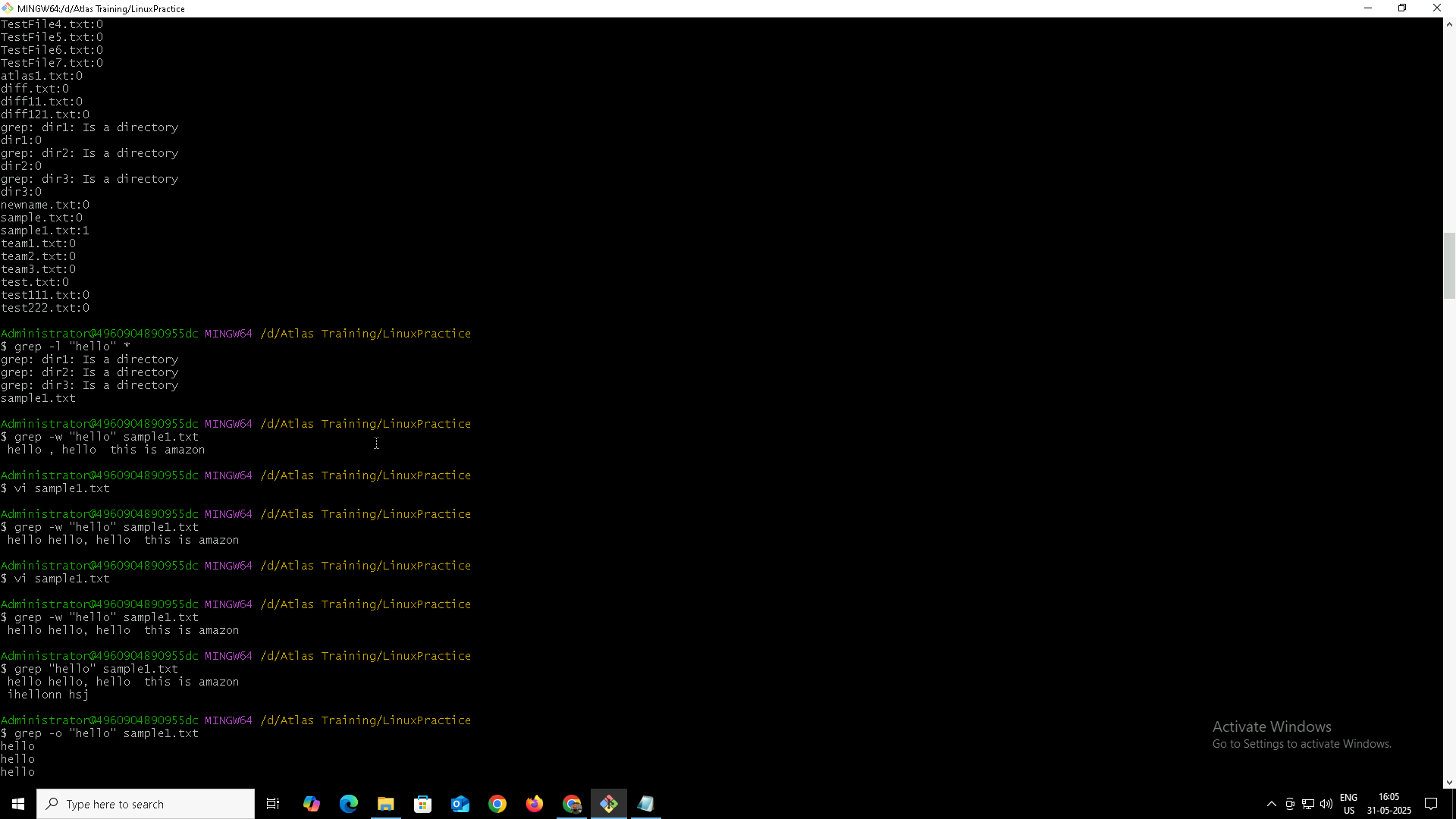
* A smaller unit within a process sharing the same memory.
* Lightweight, faster to create.
* Crashes may affect the entire process.

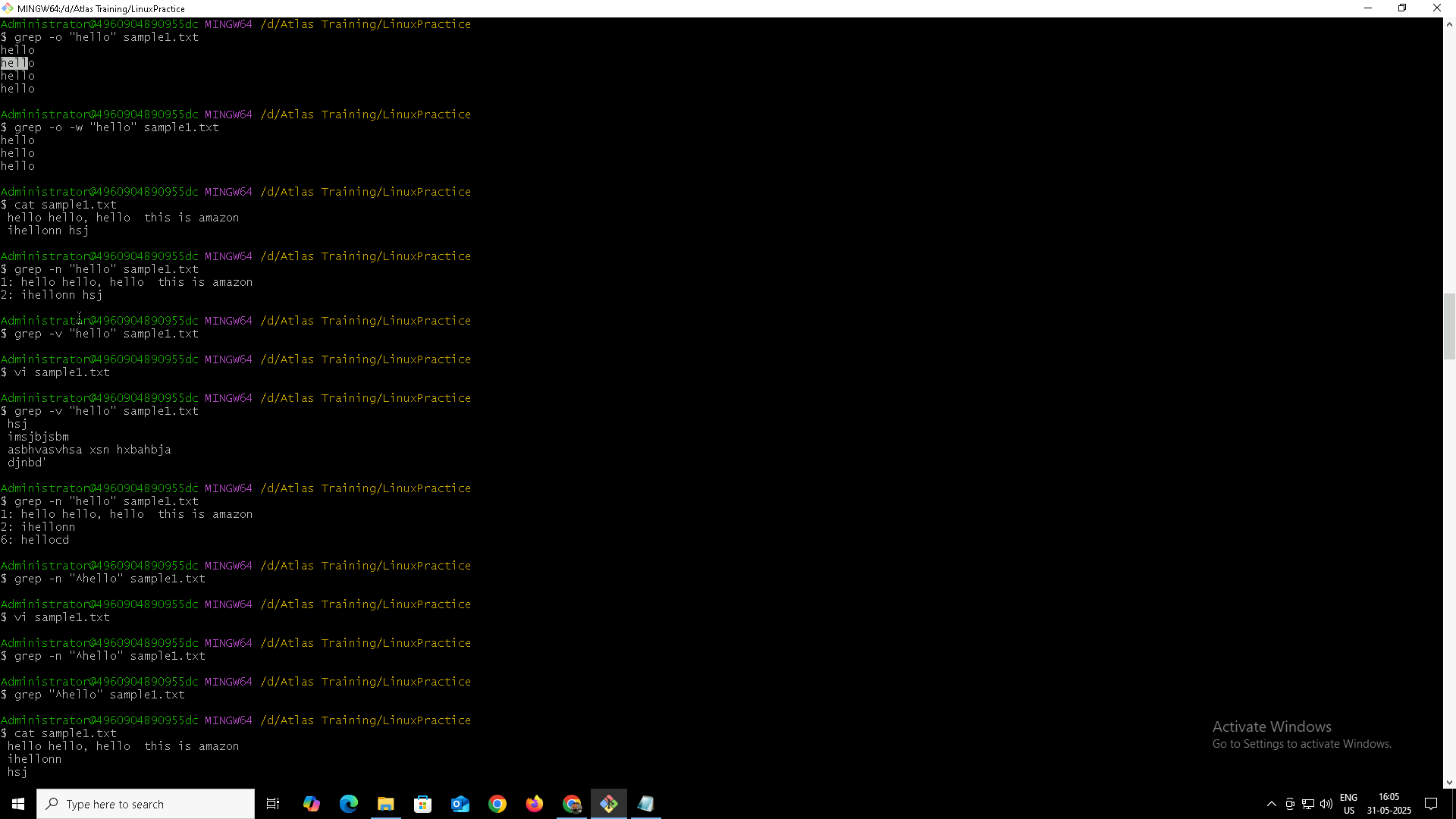
In short, a process is like a full program, and threads are tasks inside that program running simultaneously.

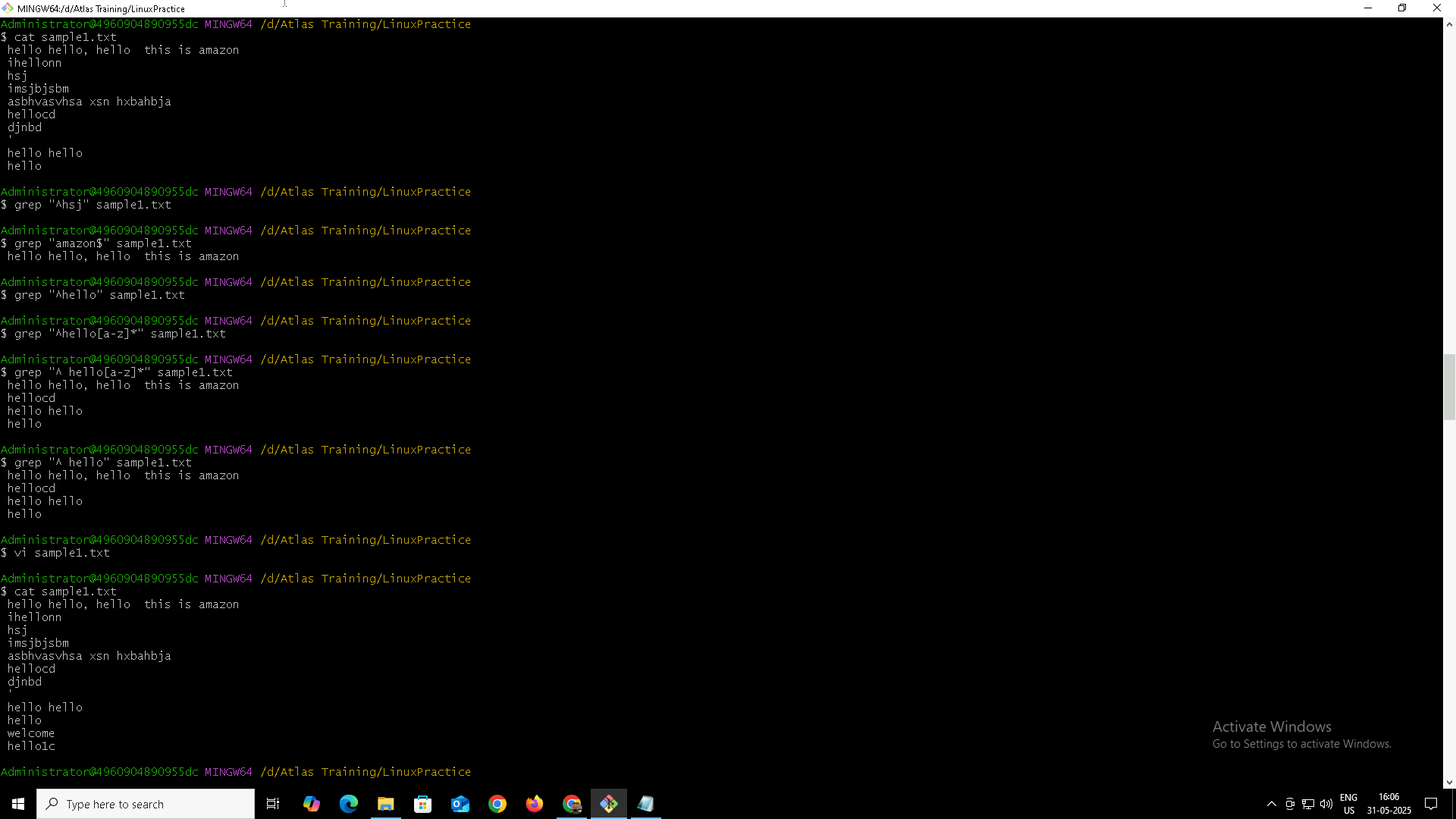
**Task 16**

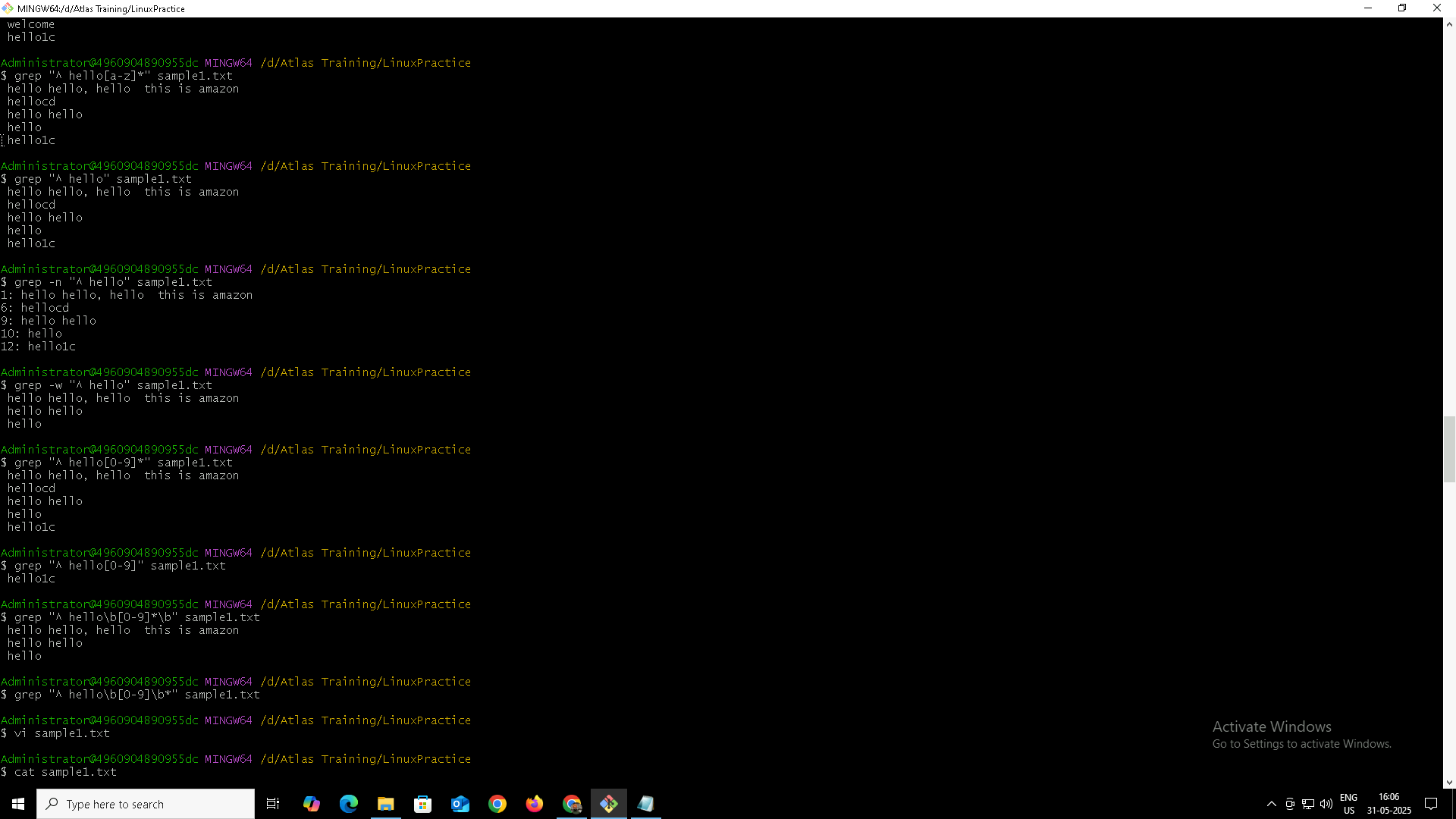
**Linux grep Commands**

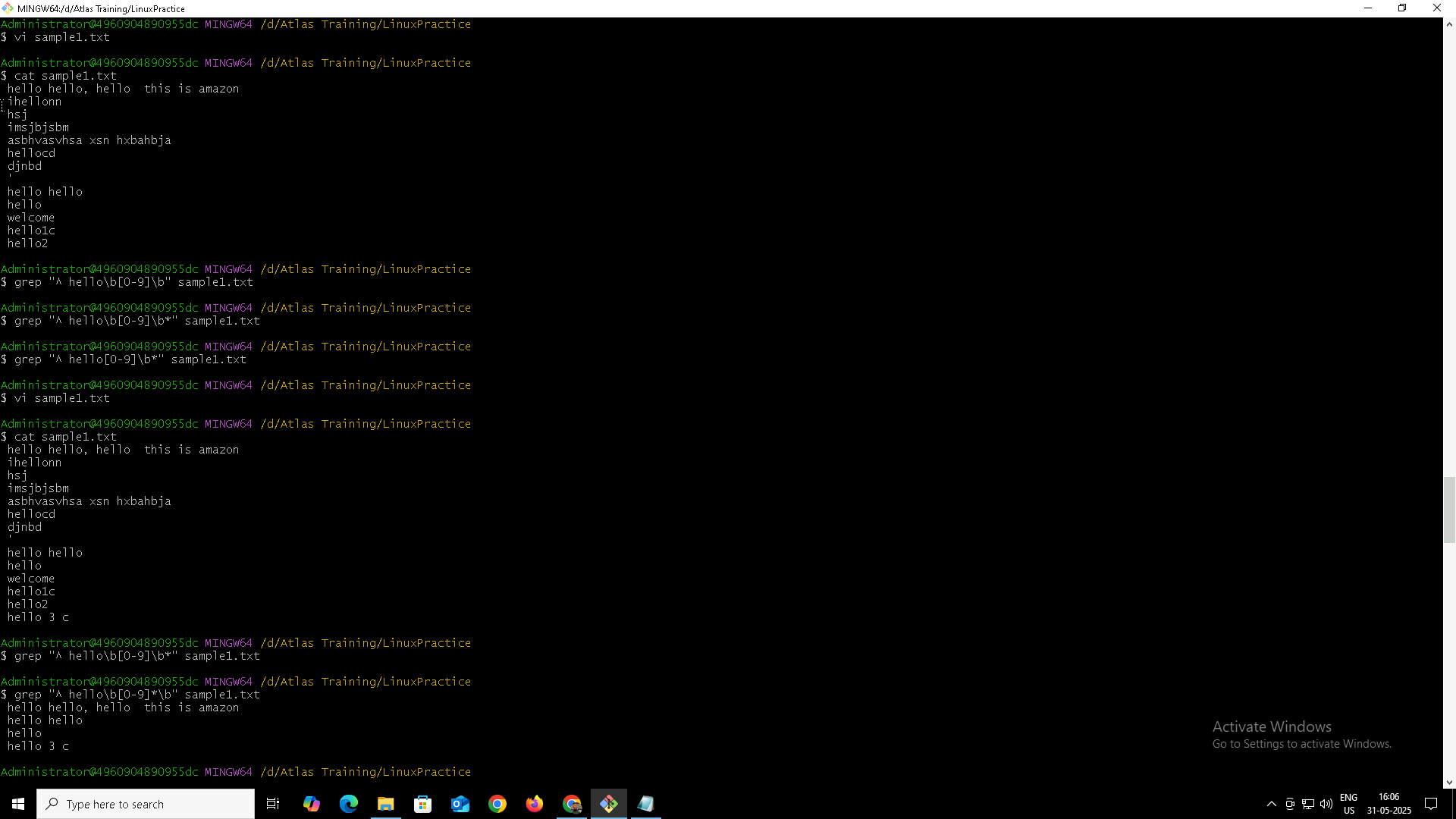
****

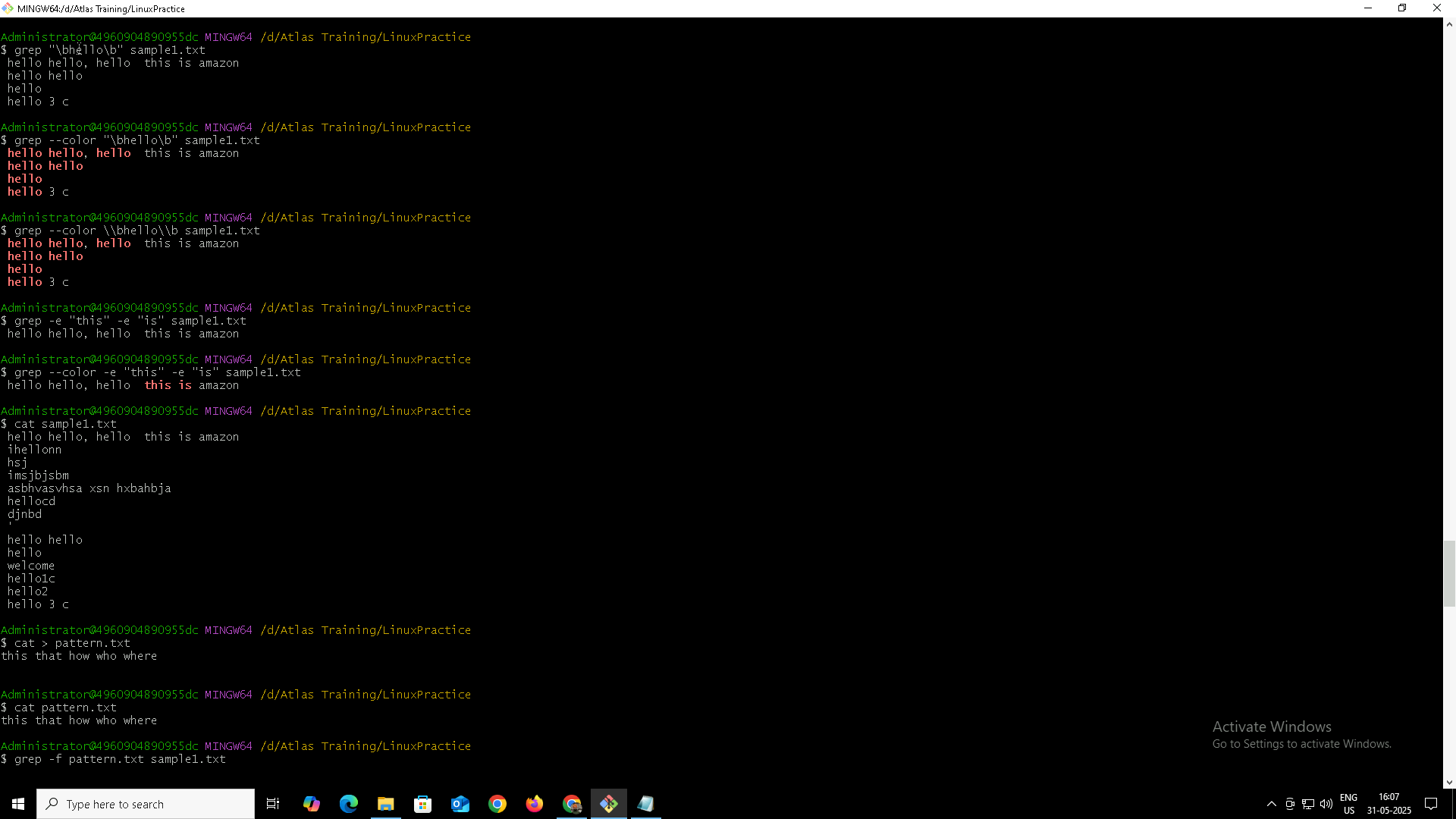
****

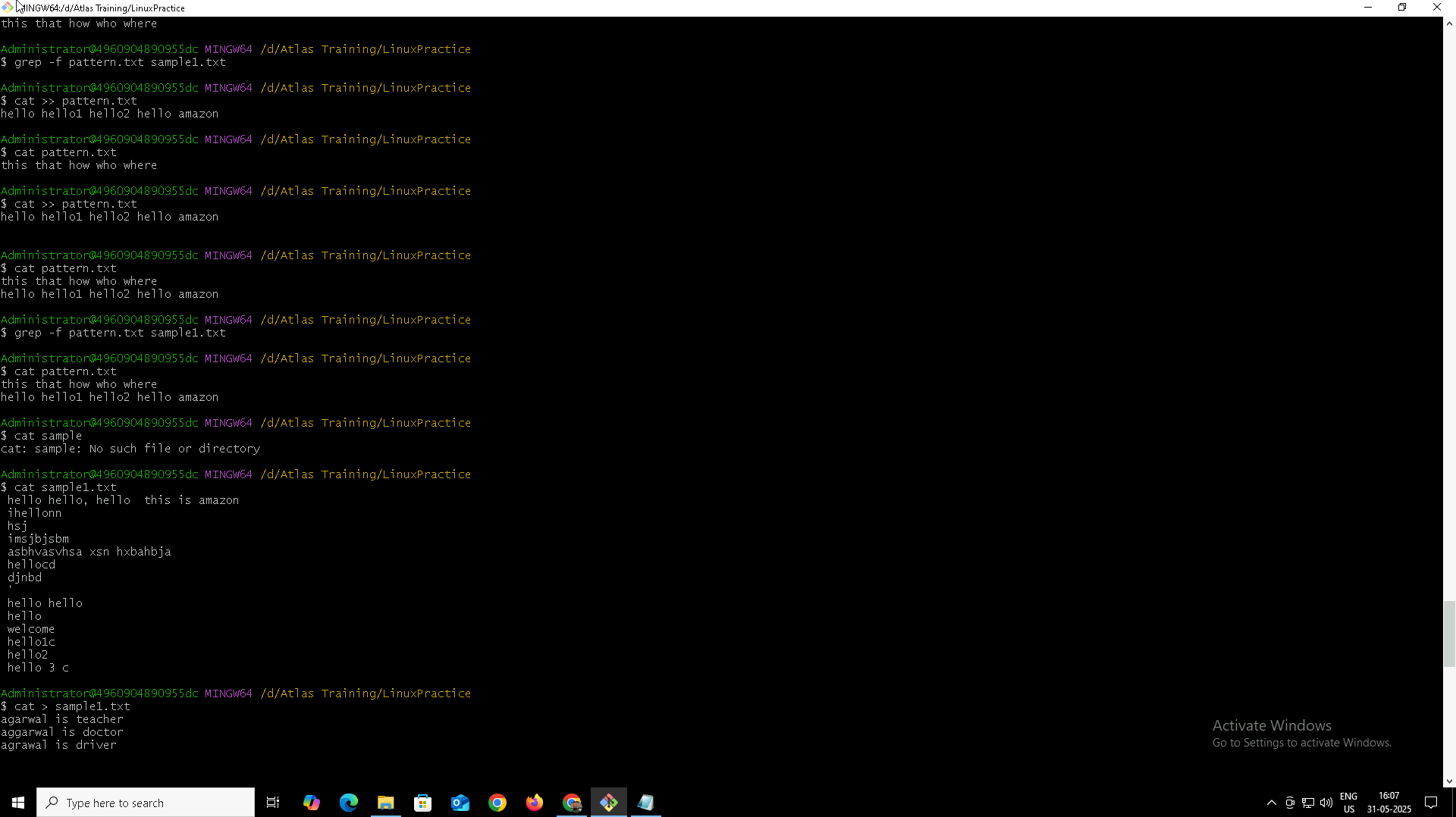
****

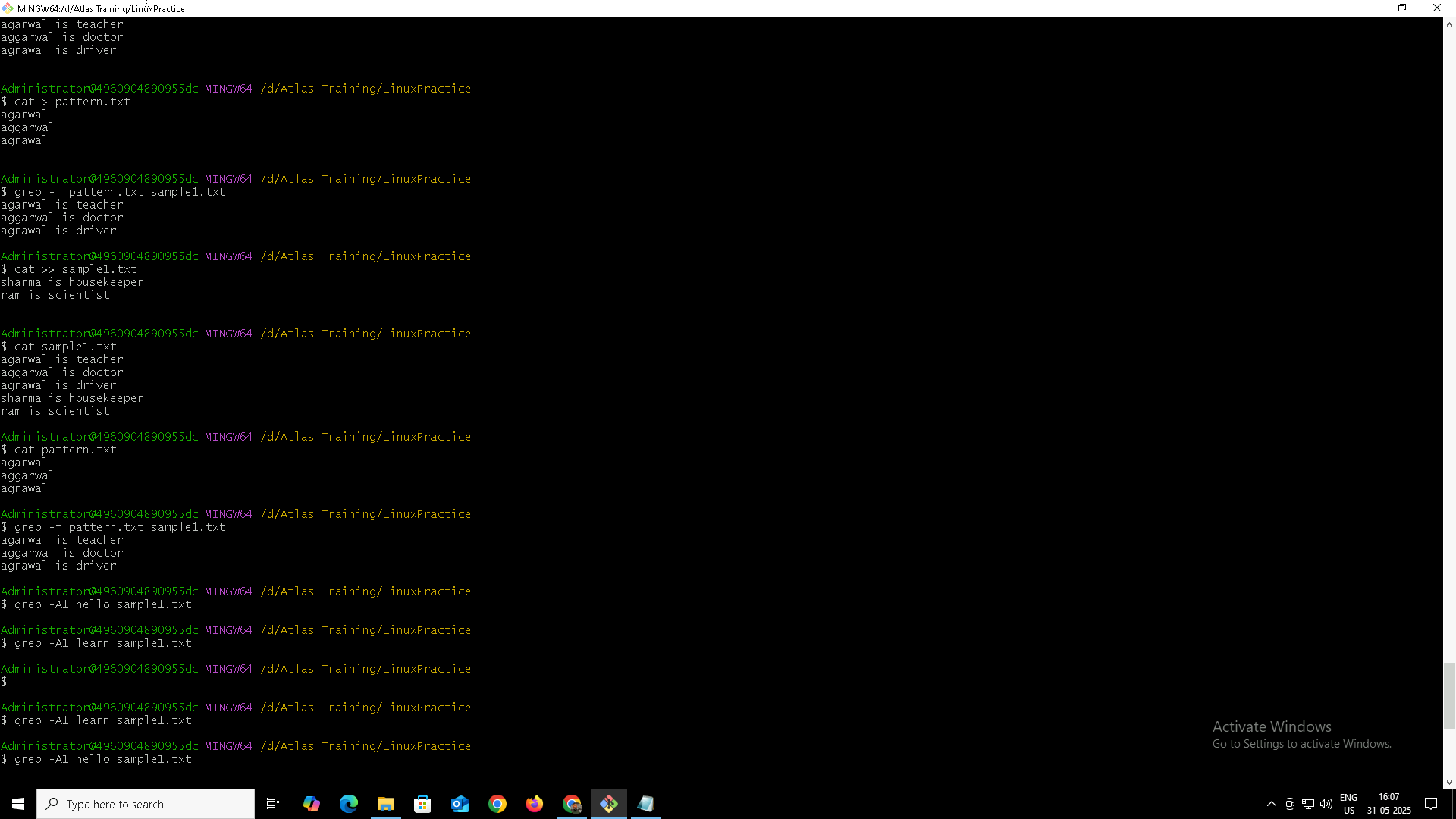
****

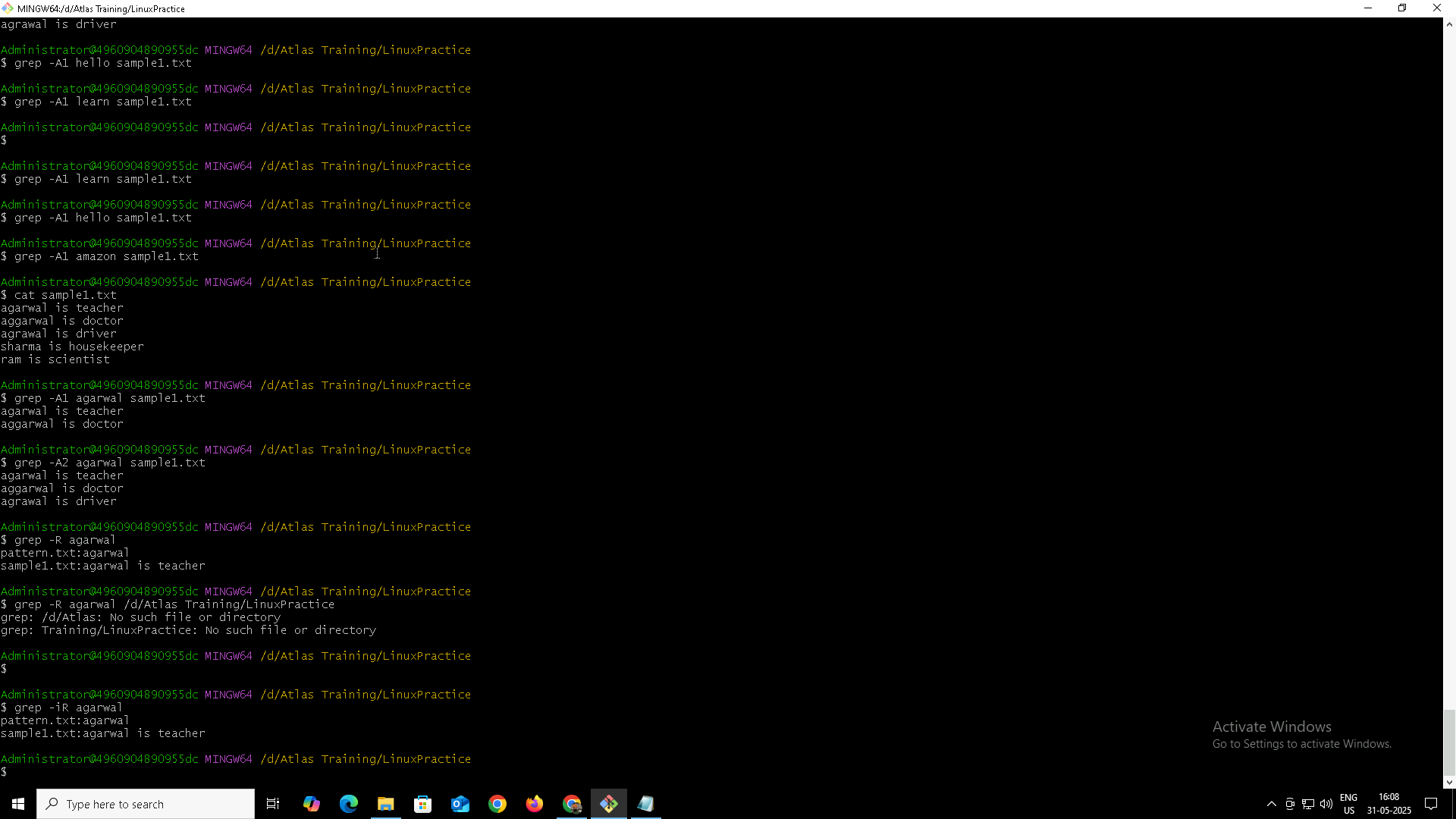
****

****

****

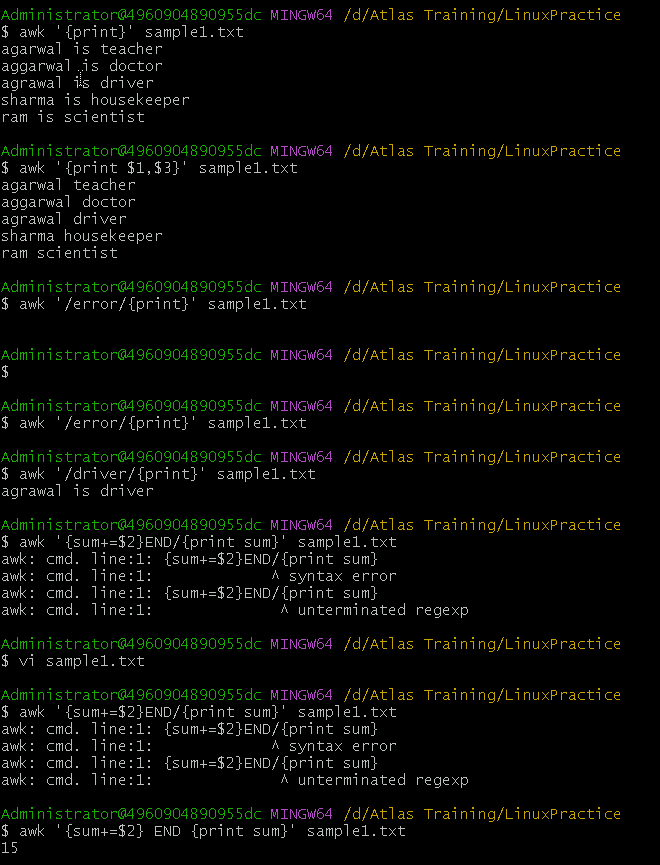
****

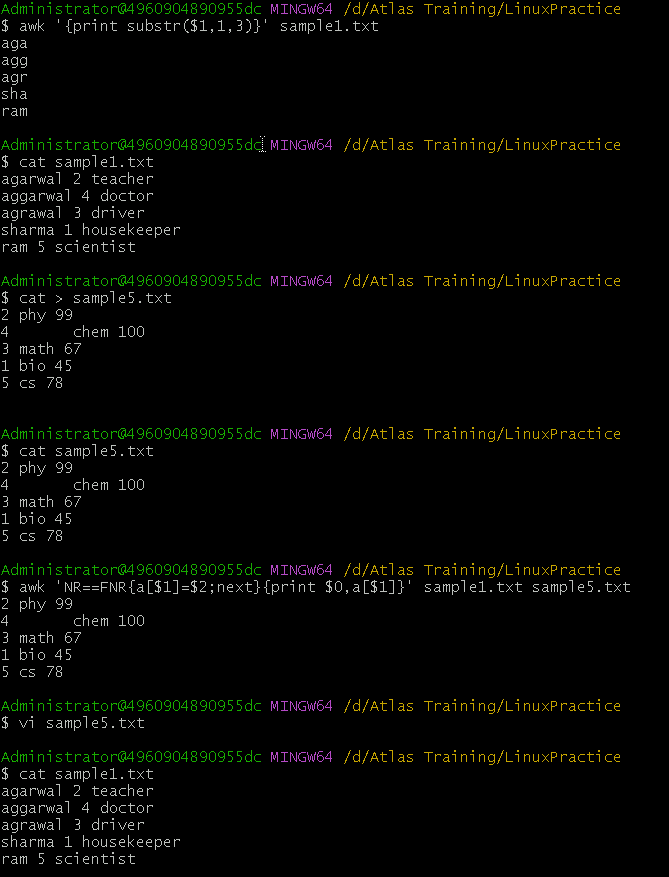
****

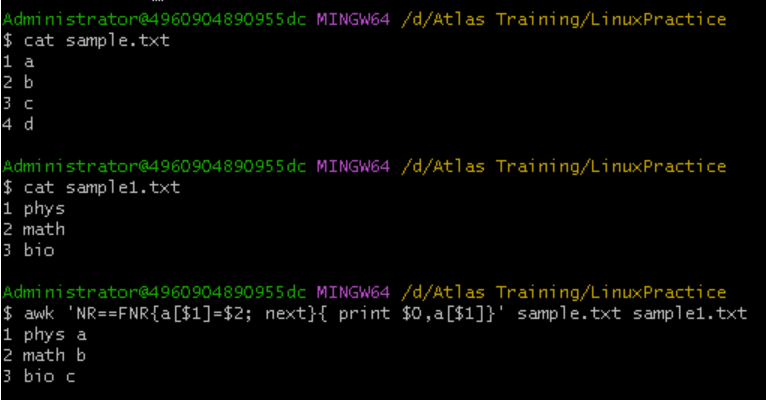
****

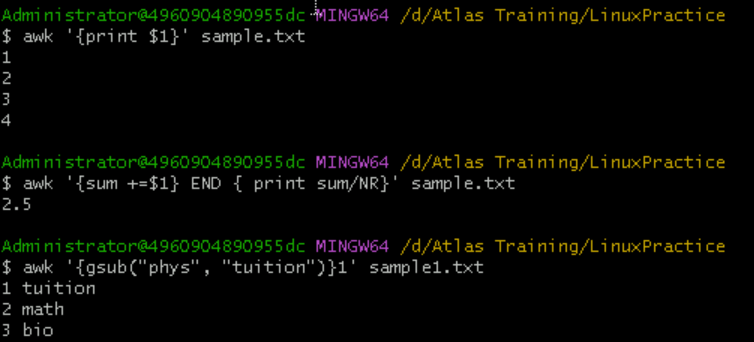
**Task 17**

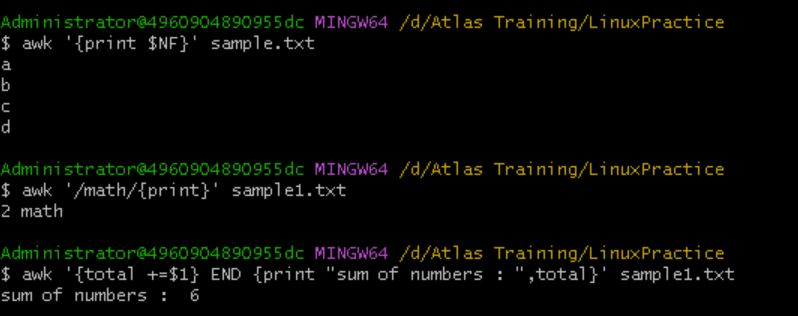
**Awk Commands**

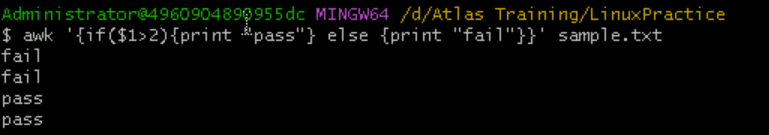
****

****

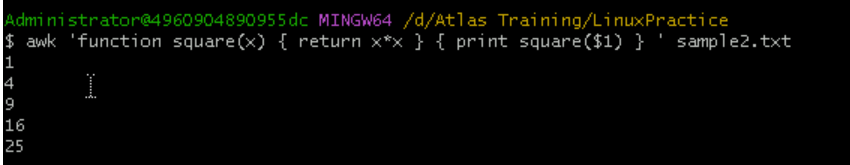
****

****

****

****

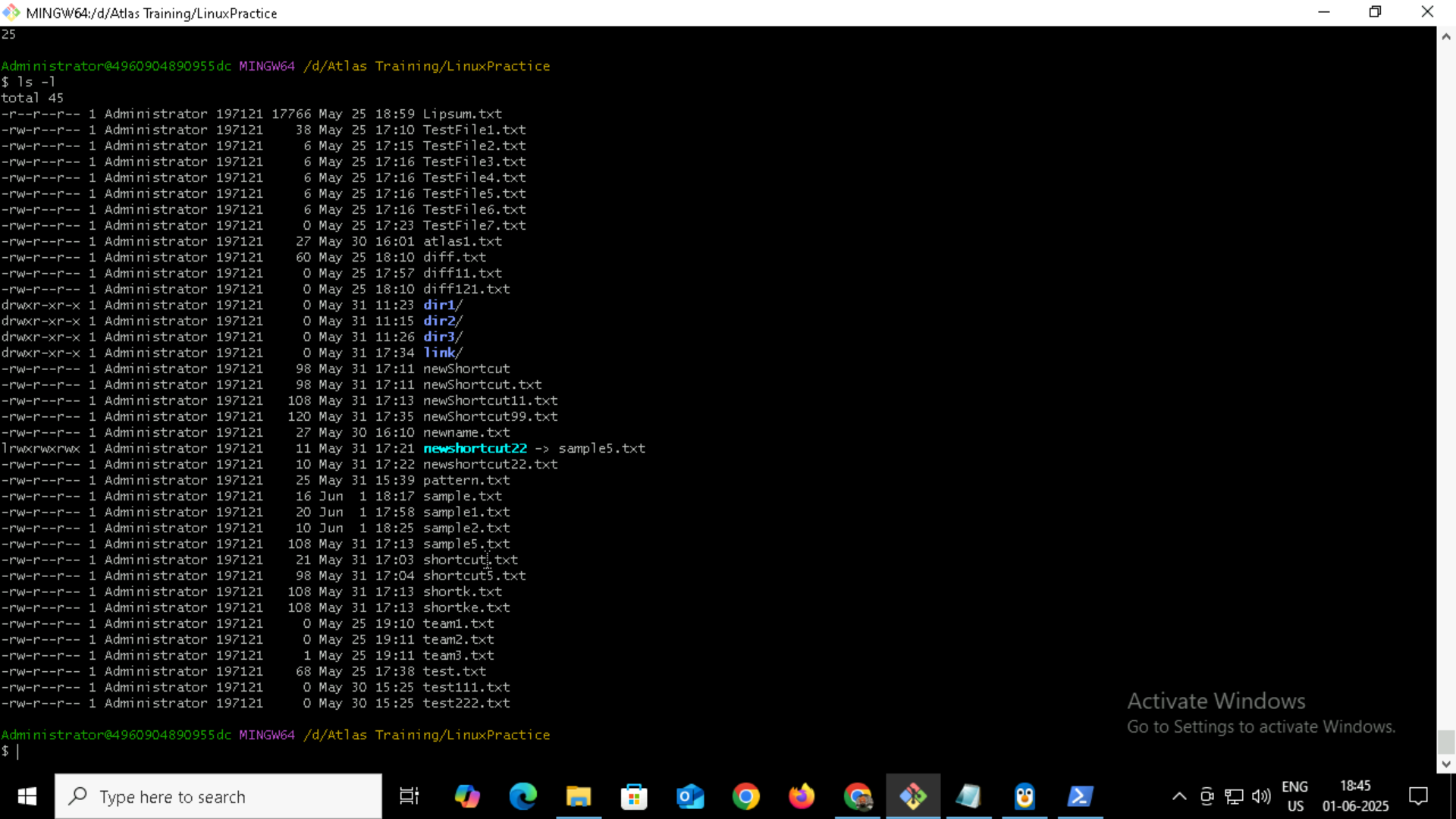
****

****

**Task 18**

**How to check file access permission in Linux**

To check **file access permissions** in Linux, we can use the ls -l command. It displays the permissions, ownership, and other details of files and directories.



**Task 19**

**Default Permission for a new file**

When you create a new file, the **default permission** is usually:

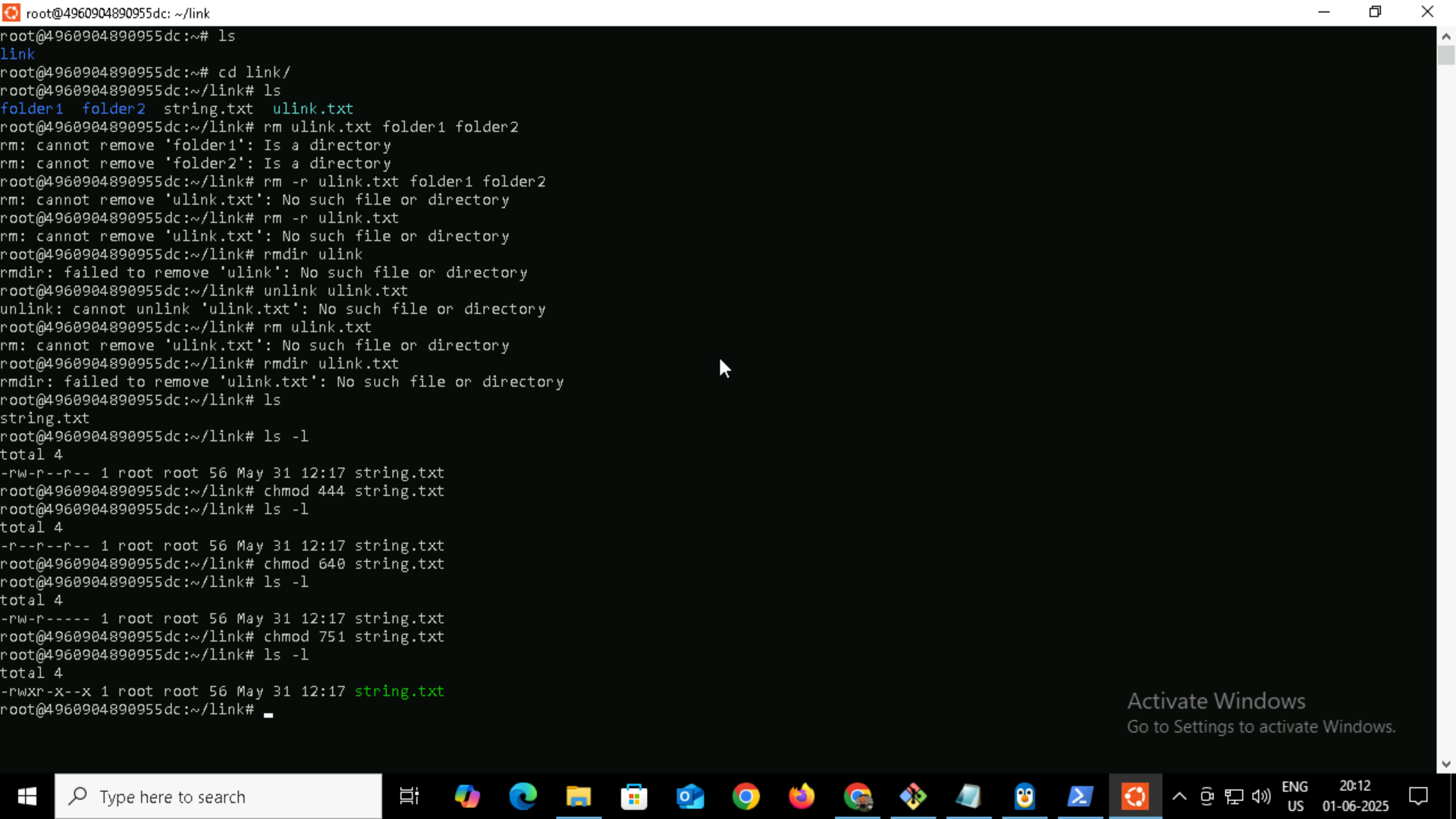
-rw-r--r–

Which means:

* **User (owner)**: read & write (rw-)
* **Group**: read only (r--)
* **Others**: read only (r--)

By default, **new files are not executable**, because:

* Most files are documents, scripts, configs — not programs.
* We must manually add execute permission if needed



**Task 20**

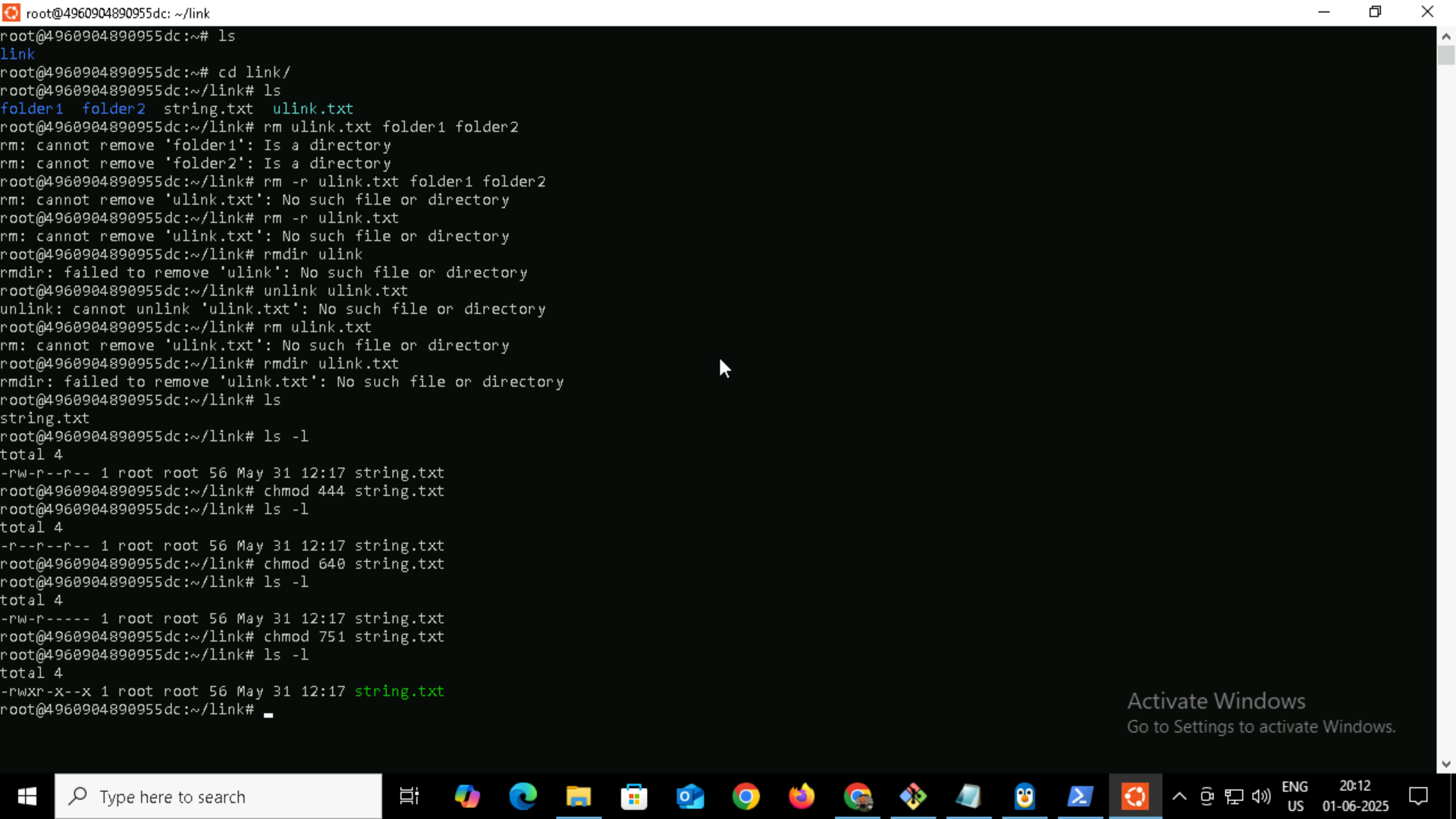
**Command to change the permission to read only for the owner, group and all other users**

To change the permission of a file so that **only read access is allowed** for:

* the **owner** (user),
* the **group**, and
* **others** (everyone else),

We can use the chmod command like this:

chmod 444 filename



**Task 21**

**Can you change the file permissions to match the following:**

* **owner: Read and Write**
* **group: Read**
* **other: no permissions (None)**

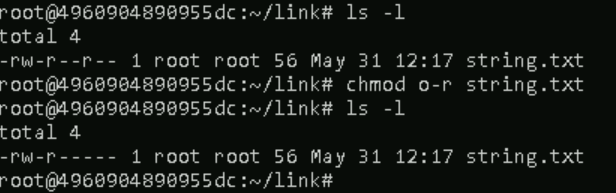
Yes, We can set those exact permissions using the chmod command.

### **Required Permissions:**

* **Owner**: Read & Write → rw- → 6
* **Group**: Read only → r-- → 4
* **Others**: No permission → --- → 0

### **Command:**

chmod 640 filename



**Task 22**

**Command for changing the file permissions to -rw-r-----?**

Breakdown of -rw-r-----:

Owner Read + Write (rw-) 4+2

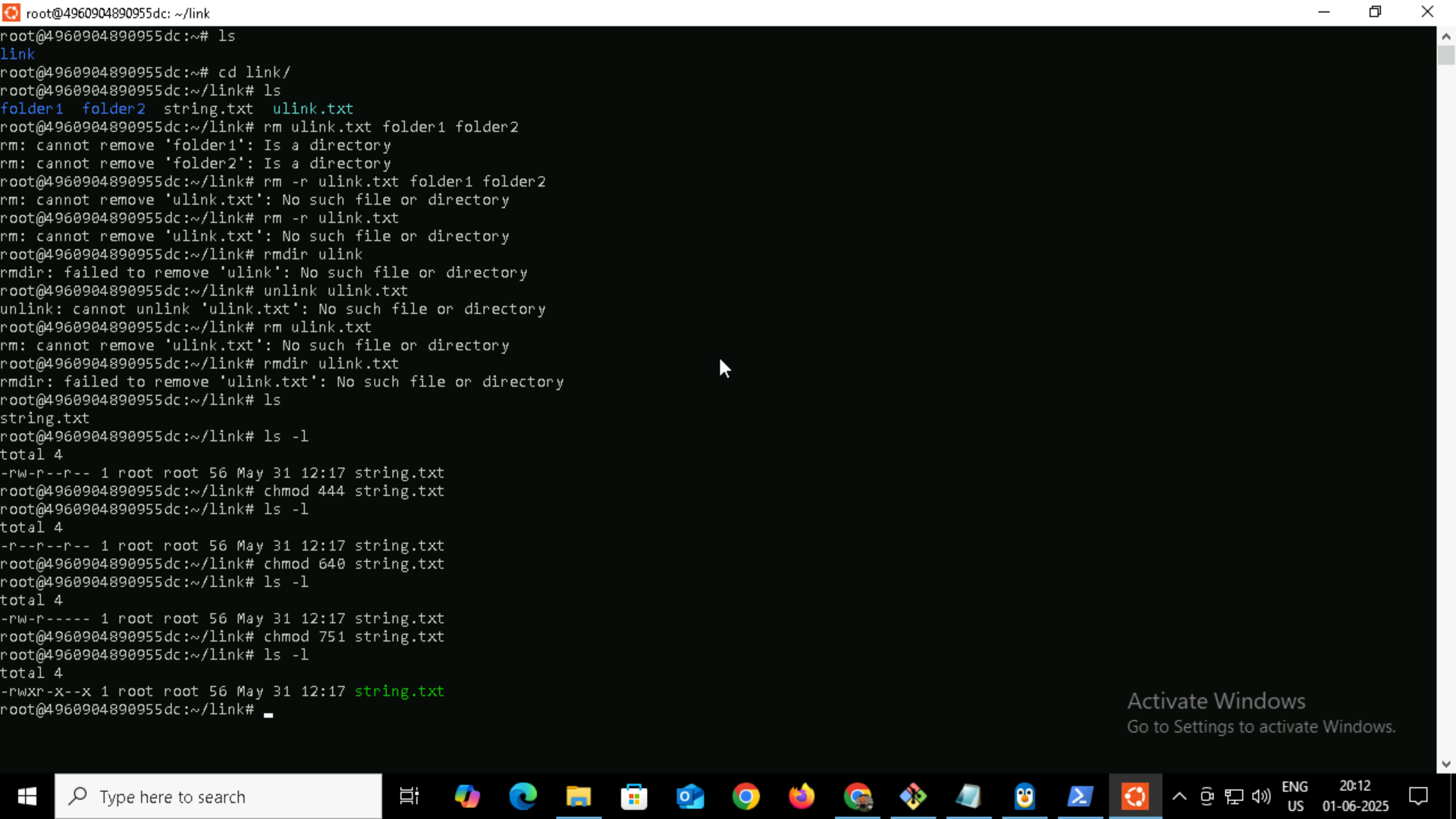
Group Read only (r--) 4

Others No permission (---) 0

So chmod 640 = **6 (owner) + 4 (group) + 0 (others)**

**And Command will be :**

chmod 640 filename



**Task 23**

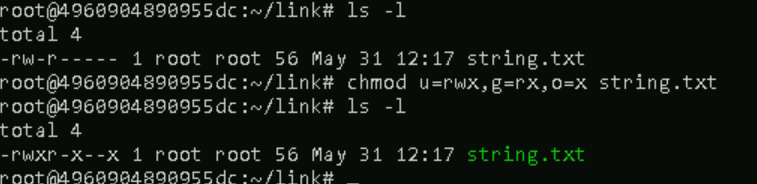
**Can you change the file permissions to match the following:**

* **owner: Read, Write and Execute**
* **group: Read and Execute**
* **other: Execute**

To set the file permissions as described:

* **Owner**: Read, Write, Execute → rwx → **7**
* **Group**: Read, Execute → r-x → **5**
* **Others**: Execute only → --x → **1**

**Command**  **:**  chmod 751 filename



**Task 24**

**Command for changing the file permissions to -rwxr-x--x?**

Breakdown of -rwxr-x--x:

Owner Read + Write + Execute(rwx) 4+2+1

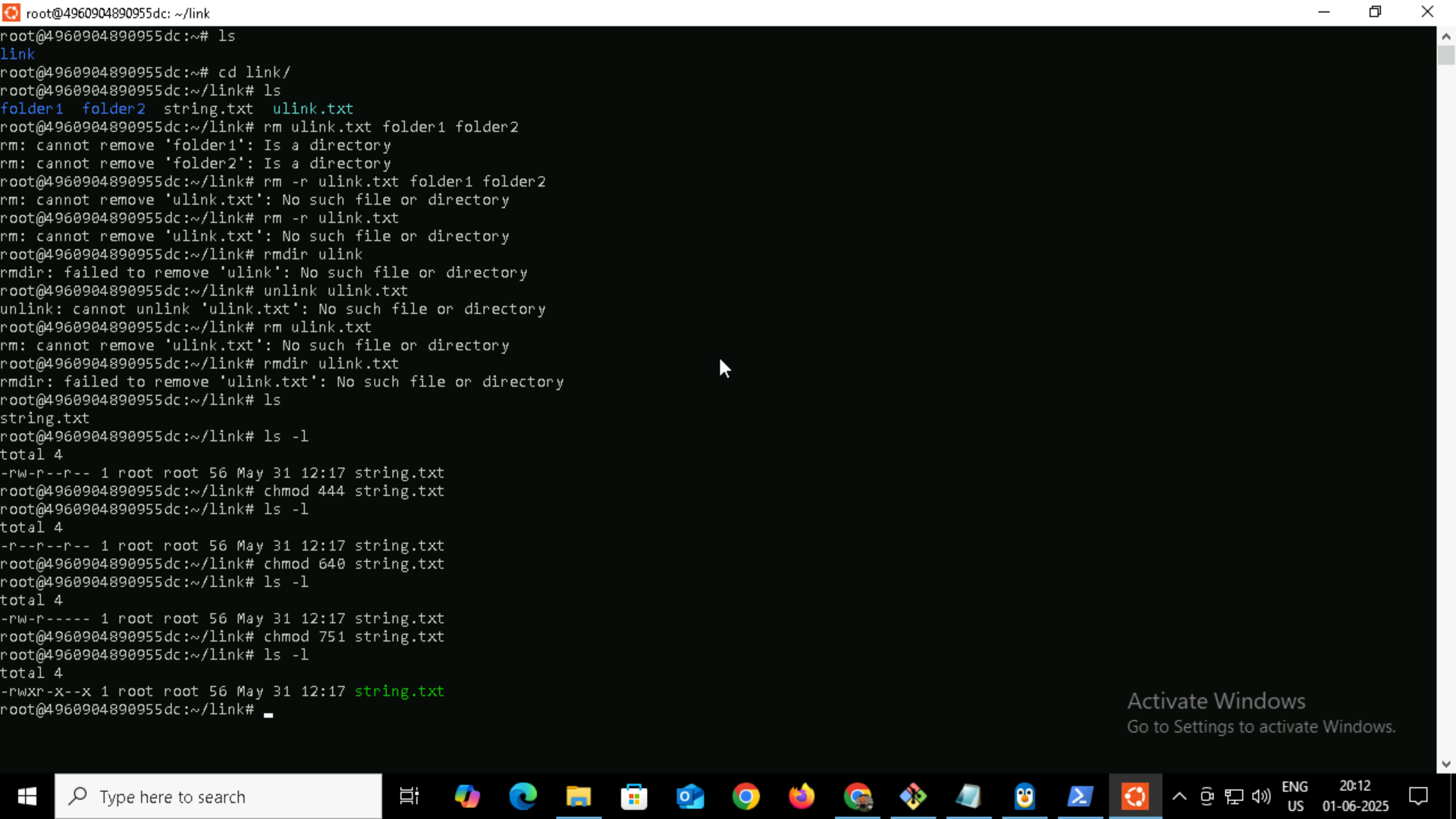
Group Read + Execute (r-x) 4+1

Others Execute Only (--x) 1

So chmod 640 = **7 (owner) + 5 (group) + 1 (others)**

**And Command will be :**

chmod 751 filename



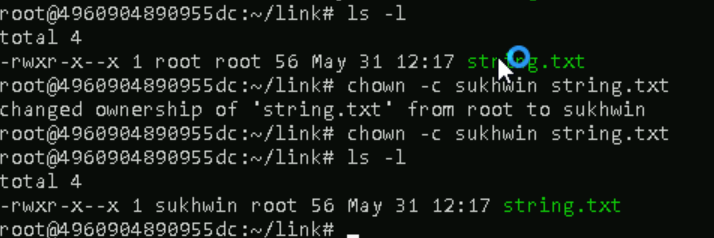
**Task 25**

**What “chown -c master file1.txt” command will do?**

It will :

### Change the ownership of file1.txt to the user master, and

### Display a message only if a change was made, because of the -c (--changes) flag.



**Task 26**

**What is a Process?**

A **process** is an instance of a program that is currently being executed.

### In simple terms:

* When you run a command or program, Linux creates a process.
* It has its own memory, unique Process ID (PID), and runs independently.

**Task 27 and Task 28**

**Command to check foreground process and background process and list all the running processes?**

Commands to check foreground and background processes in Linux:

### **1. Foreground Process:**

* A foreground process is one that is currently attached to the terminal and actively running.

sleep 30

This will run sleep in the foreground — and we won't be able to type in the terminal until it finishes or you stop it.

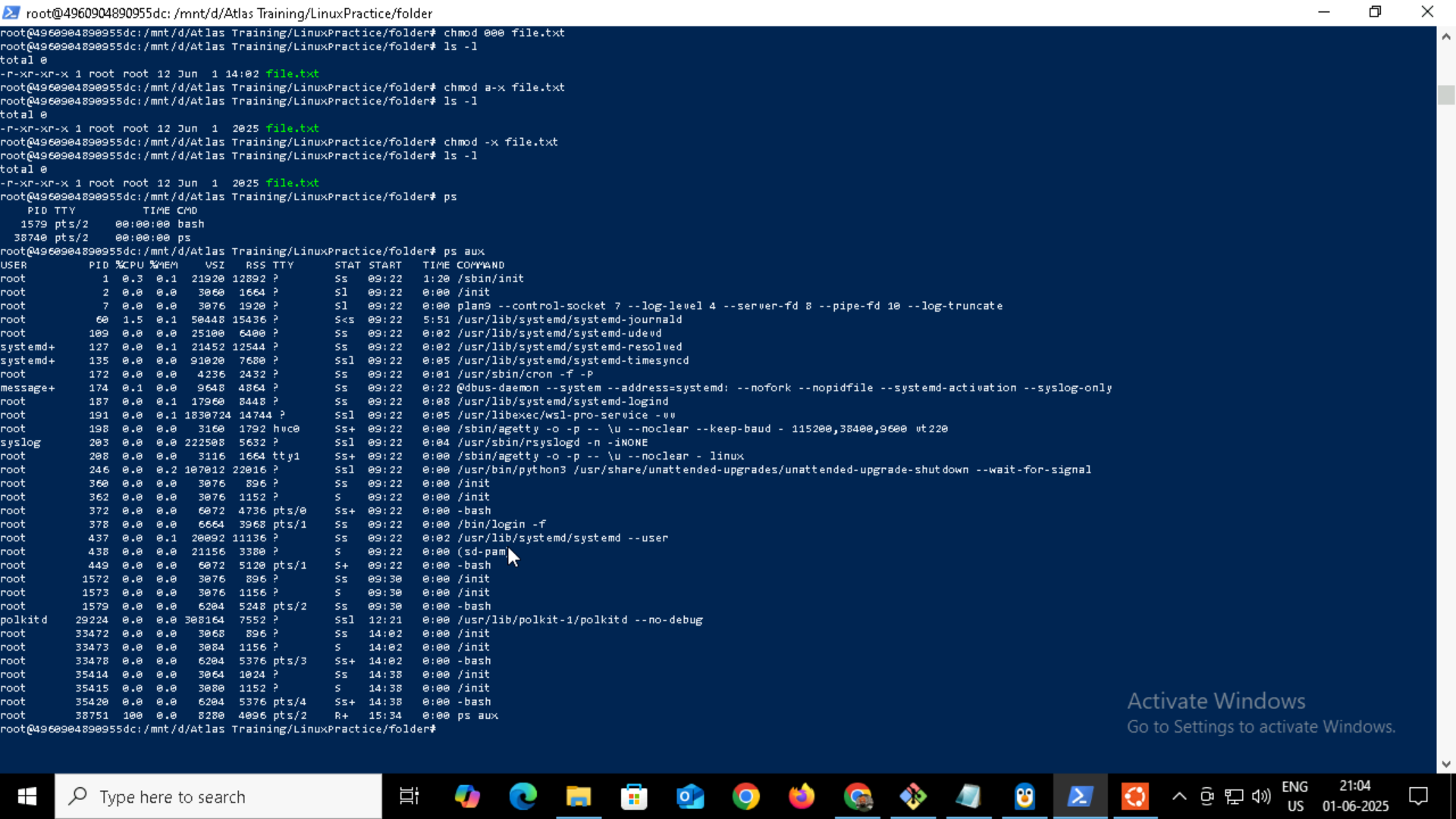
### **2. Background Process:**

* We can start a process in the background by appending &:

sleep 60 &

This starts sleep in the background, and the shell will show a job ID and PID.

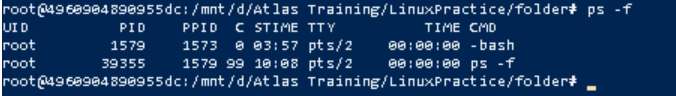
Ps and ps aux



**Task 29**

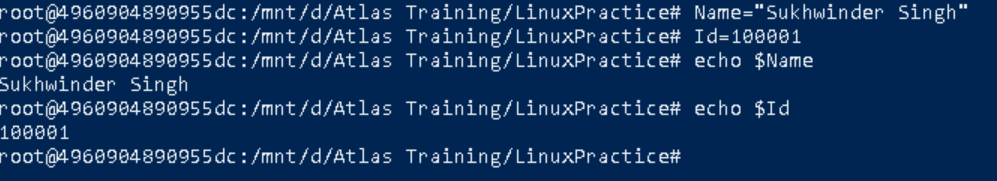
**What will ps -f command do ?**

It is used to display **currently running processes** in a **full-format listing** (more detailed than the default).



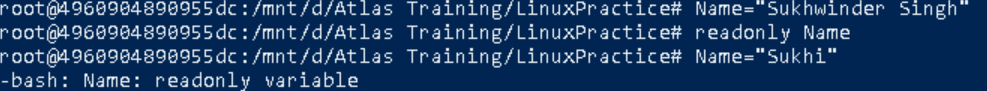
**Task 30**

**Creating a variable Name and Id with our name**



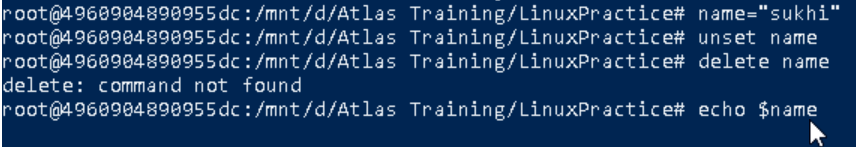
**Task 31**

**Making the above variable readonly**

****

**Task 32**

**Unset or delete the variable and specify the reason to print empty line.**

****

### Empty line comes after unsetting and echoing the same variable(name).

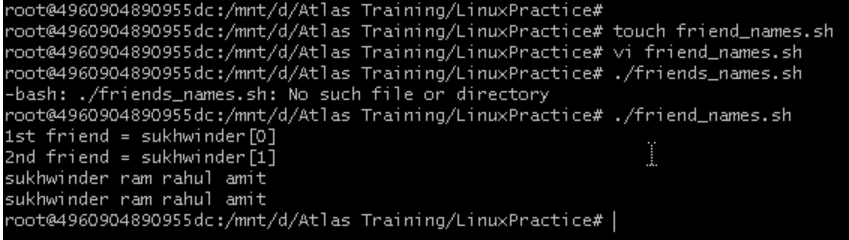
### **Because :**

* unset name removes the variable name from the shell's environment.
* So when we do echo $name, there's nothing to expand, resulting in no output (just a blank line).

**Task 33 and Task 34**

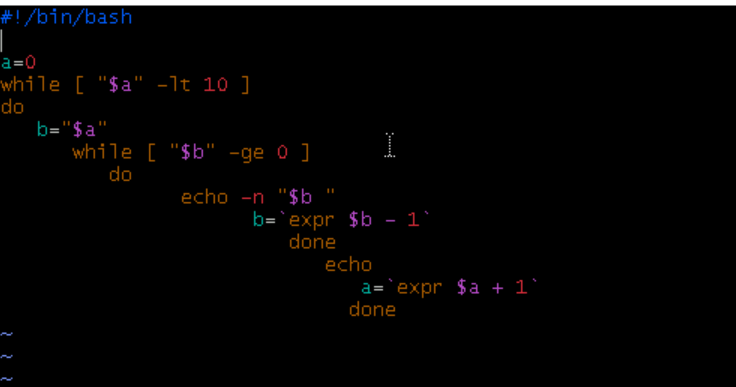
**Try to add a list of friend’s names in an array and print it and can we print the list at once in an array**

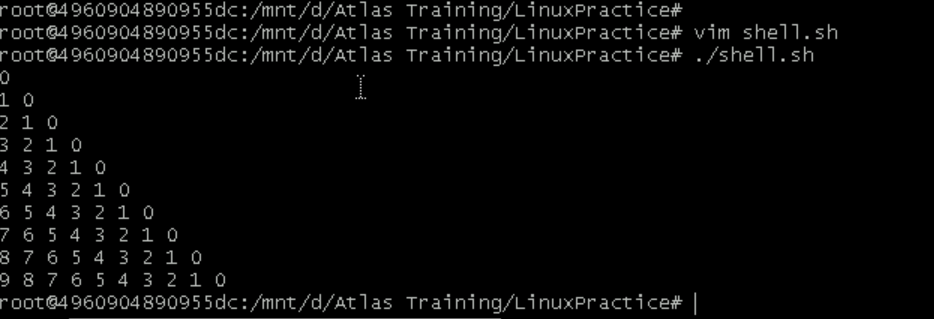




**Task 35**

**What’s the output of the below snippet:**





**Symbolic links**



