**TASK - 1**

RegEX Symbols in linux

List them down with description

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| . | Matches **any single character** except a newline |
| ^ | Matches the **start of a line** |
| $ | Matches the **end of a line** |
| \* | Matches **zero or more** occurrences of the previous character |
| + | Matches **one or more** occurrences of the previous character (Extended RegEx) |
| ? | Matches **zero or one** occurrence (Extended RegEx) |
| \ | Escapes a special character (e.g., \. matches a dot) |
| [] | Matches **any one** character inside the brackets (character class) |
| [^] | Matches any character **not** inside the brackets |
| ` | ` |
| () | Groups expressions together — Extended RegEx |

### 

|  |  |
| --- | --- |
|  |  |
| [abc] | Matches any one character: a, b, or c |
| [a-z] | Matches any one **lowercase letter** |
| [A-Z] | Matches any one **uppercase letter** |
| [0-9] | Matches any one **digit** |
| [a-zA-Z0-9] | Matches any alphanumeric character |

### **⏳ Quantifiers (Extended RegEx)**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| {n} | Exactly **n** occurrences |
| {n,} | **n or more** occurrences |
| {n,m} | Between **n and m** occurrences |

### **📌 Special Sequences (with grep -P or perl)**

|  |  |
| --- | --- |
| **Symbol** | **Description** |
| \d | Digit (same as [0-9]) |
| \D | Non-digit |
| \w | Word character (letters, digits,\_) |
| \W | Non-word character |
| \s | Whitespace character |
| \S | Non-whitespace character |
| \b | Word boundary |
| \B | Not a word boundary |

**TASK - 2**

If you are aware of Linux OS.. can you tell me the feature of Linux.

Linux is a powerful, open-source operating system that powers everything from smartphones to supercomputers. Here are its key features:

### **Key Features of Linux:**

#### **1. Open Source**

* Source code is freely available.
* Anyone can view, modify, and redistribute it.
* Encourages collaboration and community support.

#### **2. Multitasking**

* Can run multiple tasks/processes at the same time without slowing down the system.

#### **3. Multi-user**

* Multiple users can use the system simultaneously, each with their own permissions and environment.

#### **4. Portability**

* Linux runs on a wide range of hardware, from servers to embedded devices.

#### **5. Security**

* Built-in security features like file permissions, SELinux, and firewalls.
* Less vulnerable to viruses compared to other OSes.

#### **6. Stability and Reliability**

* Rarely crashes or needs to reboot.
* Uptime can last months or years without issues.

#### **7. Modular and Customizable**

* Linux is made up of modules — you can customize the kernel, install only the software you need.
* Perfect for embedded systems or minimal installations.

#### **8. Command-Line Interface (CLI)**

* Powerful and flexible.
* Scripting and automation make Linux a favorite for sysadmins and developers.

#### **9. File System Support**

* Supports various file systems: ext4, XFS, Btrfs, FAT, NTFS, etc.

**TASK - 3**

What is Kernal ? can you explain about it in your words..

The kernel is the core part of an operating system. Think of it as the "brain" or "engine" that sits between your hardware and software.

Kernel acts as a bridge between hardware and software.

1. **Manages Hardware**
   * Talks to your CPU, RAM, hard disk, etc.
   * Makes sure programs can use hardware safely and efficiently.
2. **Controls Processes**
   * Starts, stops, and switches between programs (processes).
   * Manages multitasking.
3. **Manages Memory**
   * Allocates RAM to different programs.
   * Makes sure they don’t interfere with each other.
4. **Handles Input/Output (I/O)**
   * Manages devices like keyboard, mouse, printer, USB, etc.
5. **Ensures Security & Access Control**
   * Prevents apps/users from doing unauthorized actions.
   * Protects data using permissions.

### **Example:**

When you type something in a text editor and click "save":

* The application asks the kernel to write to the hard drive.
* The kernel talks to the hard drive using drivers.
* The file is saved — you don’t have to know how it all worked. The kernel handled it for you.

**TASK - 4**

BASH in Linux full form and Explanation.

**BASH** Stands for Bourne Again Shell.

It is a command-line interpreter (shell) used in most Linux distributions.

BASH allows users to:

1. **Interact with the system** You can type commands like ls, cd, cp, etc., and the shell sends those to the system to execute.
2. **Write Shell Scripts** Automate tasks with .sh files (like mini programs).
3. **Manage files and directories** Using commands (mv, rm, mkdir, etc.).
4. **Run programs and processes** You can start, stop, and manage applications.
5. **Use programming features** BASH supports variables, loops, conditionals, and functions.

**TASK - 5**

Now that you know Linux is also an Operating System like Windows..

What do you think is the difference between Linux and Windows

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Linux** | **Windows** |
| Meaning | Linux is an open-source operating system for desktops. | Windows is not an open-source operating system. |
| Developer | Linux is developed by Linux Torvalds. | Wind  ows is developed by MICROSOFT. |
| Availability | Linux is open-source and free. | Windows is a paid operating system. |
| Usage | Linux has machine-friendly features and the user must learn to use Linux. | Windows uses Graphical User Interface and any non-technical user can use it without getting into tech-knowledge. |
| Kernel-type | Linux is based on a Monolithic kernel. | Windows is based on Microkernel. |
| Path distinguisher | Forward slash is the path distinguisher. | Backward Slash is the path distinguisher. |
| Safety | Linus possesses better safety measures than Windows. | Windows is less safe than Linux. |
| Case sensitivity | Linux is less case-sensitive. | Windows is not case-sensitive. |
| Updates | Linux updates less frequently. | Windows has frequent updates. |
| Language | Linux is written in C and Assembly language. | Windows is written in C++ and Assembly language. |
| License | Linux is initiated under General Public License. | Windows is initiated under a Proprietary commercial software license. |
| Reliability | Linux is more reliable due to its high security. | Windows is less reliable than Linux. |
| File system | Linux uses a tree structure to manage files. | Windows uses directories such as C, D, E, and more, and folders are used to store files. |

**TASK - 6**

What are the basic components of Linux? Describe each in detail with diagrams.

### **1. Hardware**

* This is the physical layer — includes the CPU, RAM, storage, keyboard, mouse, and other devices.
* Linux interacts with hardware through the kernel using device drivers.

### **2. Kernel**

The core of the Linux system

* Manages system resources (CPU, memory, devices).
* Acts as a bridge between hardware and software.
* Key functions:  
  + Process Management: Controls execution of programs.
  + Memory Management: Allocates and frees RAM.
  + File System Management: Organizes, stores, and accesses files.
  + Device Drivers: Controls hardware components.
  + Networking: Enables communication over networks.

### **3. Shell (Command Line Interface)**

The user interface to interact with the OS

* A program that takes commands from the user and gives them to the kernel to execute.
* Common shells:  
  + BASH (Bourne Again Shell) – default in many distros.
  + ZSH, KSH, CSH – alternatives with more features or different syntax.

**Example:**

**bash**

**ls -l**

**Shell interprets and tells kernel to list directory contents.**

### **4. System Libraries**

Helper code used by programs to interact with the kernel

* Provide standard functions (like opening a file, connecting to the network).
* Examples:  
  + glibc: Standard C library.
  + libm: Math library.
* These libraries are reused by applications to avoid duplicating code.

### **5. System Utilities / User Applications**

Tools and programs that make Linux usable

* Examples:  
  + cp, mv, top, df, ps, nano, vim, etc.
* These are command-line tools or GUI applications for tasks like file management, monitoring system usage, editing text, etc.

**TASK - 7**

Is it legal to edit Kernal ? when do you think we have to in case?

The Linux kernel is licensed under the GNU General Public License (GPL) v2, which means:

* We can freely use, view, modify, and distribute the kernel.
* We must release the modified version under the same license if you distribute it.
* We must provide the source code along with the binary if you share the modified version.

So, editing the kernel for personal or internal use is completely legal — even for commercial purposes.

**TASK - 8**

**LILO stands for Linux Loader.**

It is a boot loader used in Linux systems to load the operating system into memory when the computer starts.

When you power on your computer, the BIOS/UEFI firmware does basic hardware checks and then looks for a boot loader in the Master Boot Record (MBR) or EFI partition.

If LILO is installed as the boot loader, it takes over and:

1. Presents a menu (if configured) to choose between multiple OSes (Linux, Windows, etc.)
2. Loads the selected Linux kernel into RAM.
3. Hands control over to the kernel to start the OS.

**TASK - 9**

What is shell? How many shells are there and what are they ? can you explain.

A shell is a command-line interface that allows users to interact with the operating system. It interprets the commands you type and passes them to the kernel to execute.

Think of it as the middleman between the user and the system.

There are two main types:

* Command-Line Shell – text-based interface (e.g., BASH).
* Graphical Shell – GUI environments (e.g., GNOME Shell), but usually when we say "shell" in Linux, we mean command-line shell.

Types:

**BASH -** Default on most Linux distros. Powerful, widely used, supports scripting. Full form: *Bourne Again SHell*

**SH -** Original Bourne Shell. Basic, standard scripting shell. BASH is backward-compatible with it.

**ZSH -** Feature-rich, user-friendly shell with auto-completion and plugins. Popular in macOS.

**KSH** - Korn Shell – combines features of sh, csh, and other shells. Often used in enterprise systems.

**CSH** - C Shell – syntax resembles C programming language. Less common now.

**FISH** - Friendly Interactive SHell – modern shell with smart features, good for new users.

**TASK - 10**

What is swap space?

* Swap space is a special area on a hard drive or SSD that Linux uses as an extension of the computer’s physical RAM (memory).
* When RAM is full, Linux moves some inactive memory pages from RAM to the swap space.  
  This frees up RAM for active processes.
* It helps prevent out-of-memory errors and keeps your system running smoothly even under heavy load.

**TASK - 11**

Mounting is the process of making a file system accessible to the operating system by attaching it to a specific directory (called the mount point) in the directory tree.

**TASK - 12**

What is chmod command ? how to use it?

* chmod stands for change mode.
* It is a command used to change the permissions (read, write, execute) of files and directories in Linux.
* Permissions control who can read, write, or execute a file or directory.

Syntax : chmod [who][operator][permission] filename

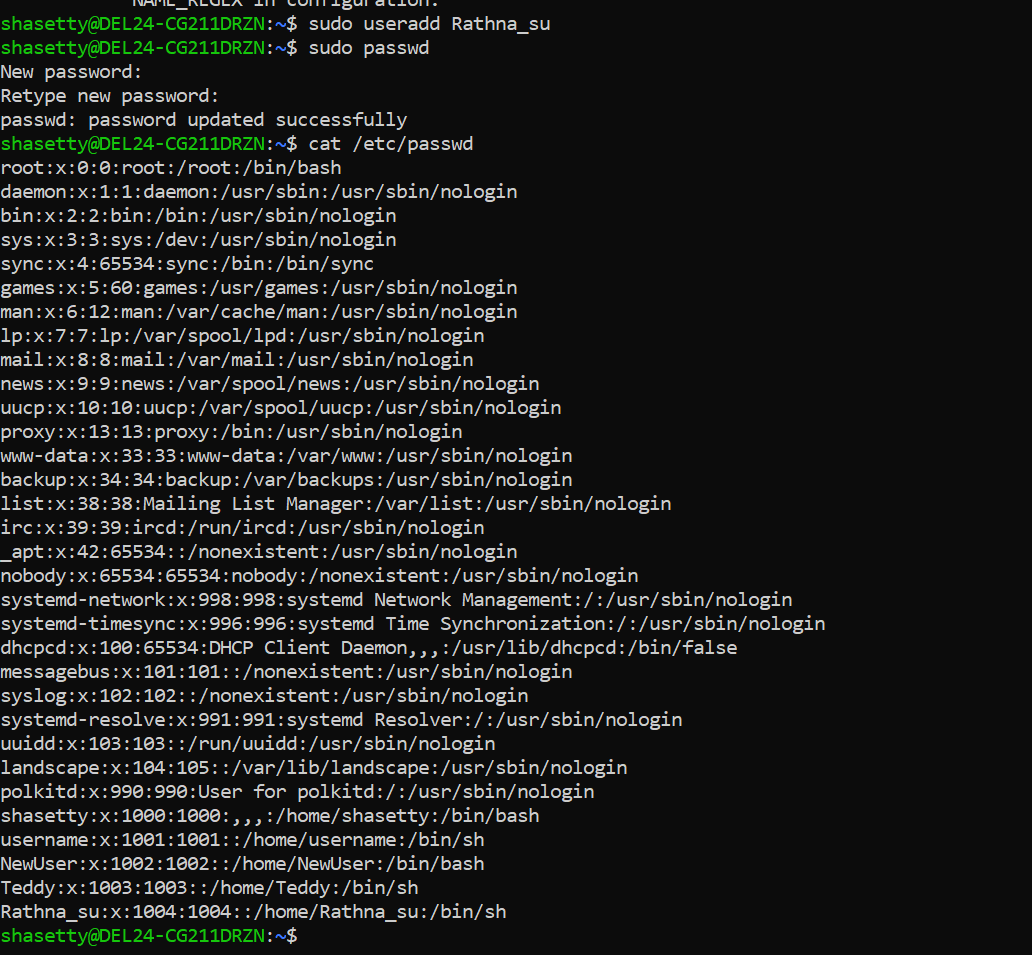
Add execute permission for user : chmod u+x script.sh

Remove write permission for group : chmod g-w file.txt

Set read and write permission for all the users : chmod a=rw file.txt

**TASK - 13**

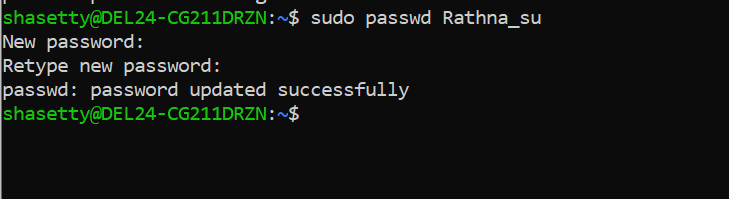
Can you add a new user account? Crate a new user in different ways and paste ss



**TASK – 14**

Can you change the password of a user?

How do you do that? Plz share ss



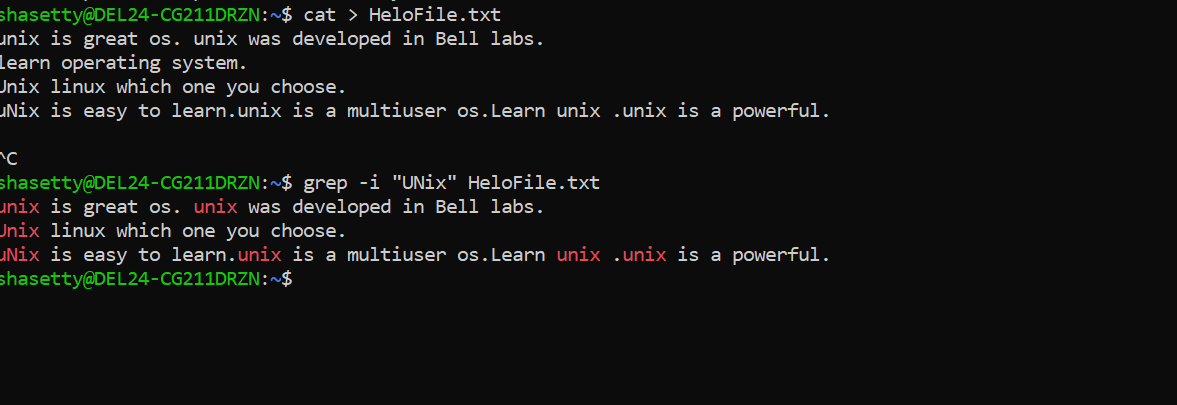
**TASK – 15**

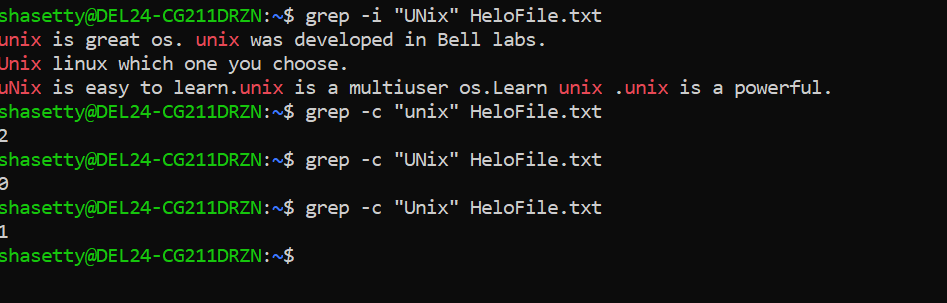
What is diff between Process and Thread?

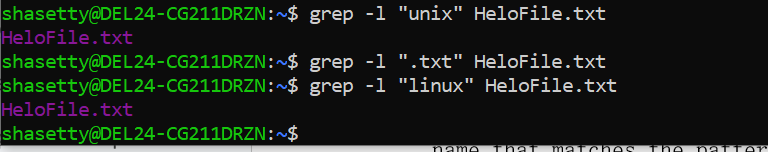
| **Aspect** | **Process** | **Thread** |
| --- | --- | --- |
| **Definition** | An independent unit of execution containing its own memory space. | A lightweight, executable unit within a process. |
| **Memory Space** | Has its own separate memory space. | Shares memory space with other threads within the same process. |
| **Overhead** | Higher overhead due to separate memory and resource allocation. | Lower overhead, more efficient in resource usage. |
| **Execution** | Operates independently. | Depends on the process; multiple threads can run in parallel within one process. |
| **Communication** | Communicates through IPC mechanisms. | Communicates directly through shared memory. |
| **Control** | Can be started, stopped, and controlled independently. | Controlled within the context of a process. |
| **Resource Allocation** | Each process has its own resources (files, variables, etc.). | Share resources of the process they belong to. |
| **Isolation** | Processes are isolated from each other. | Threads can directly affect each other within the same process. |
| **Failure Impact** | Failure in one process does not affect other processes. | A failure in one thread can affect all threads of the process. |
| **Creation Time** | Longer creation time due to resource allocation. | Shorter creation time since less resource allocation is needed. |
| **Use Case** | Suitable for applications needing isolated execution. | Ideal for tasks requiring frequent communication and shared resources. |

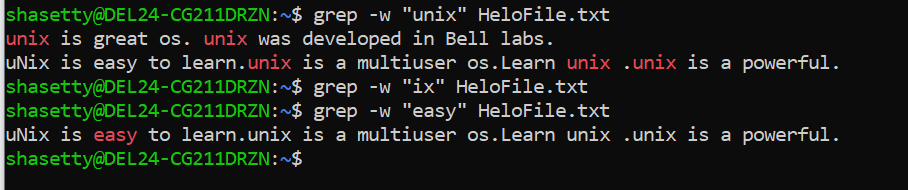
**TASK – 16**

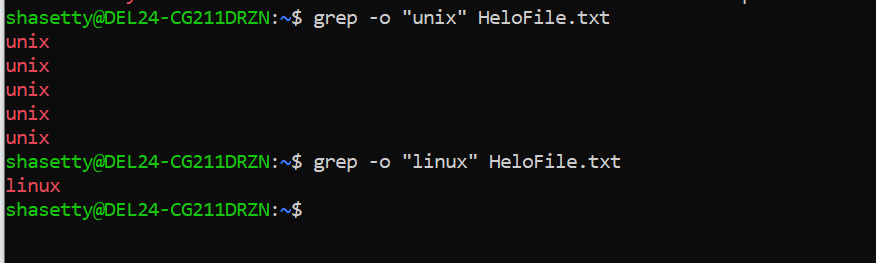
**GREP Commands**

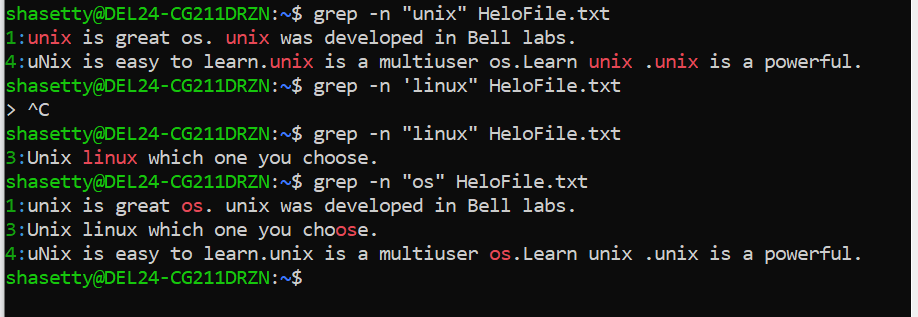


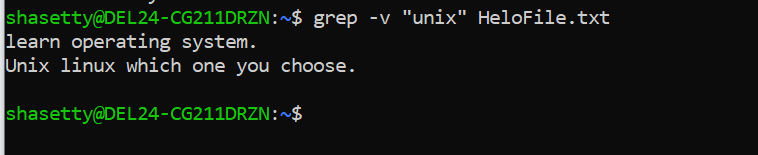


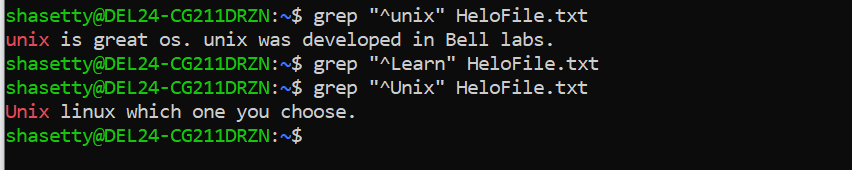


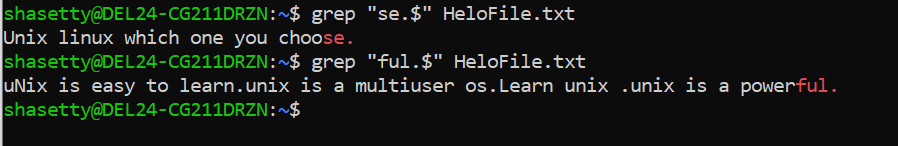


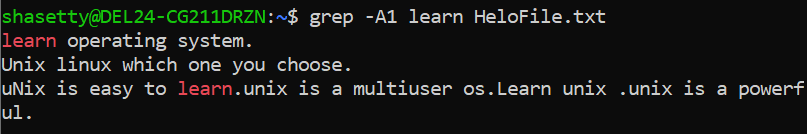






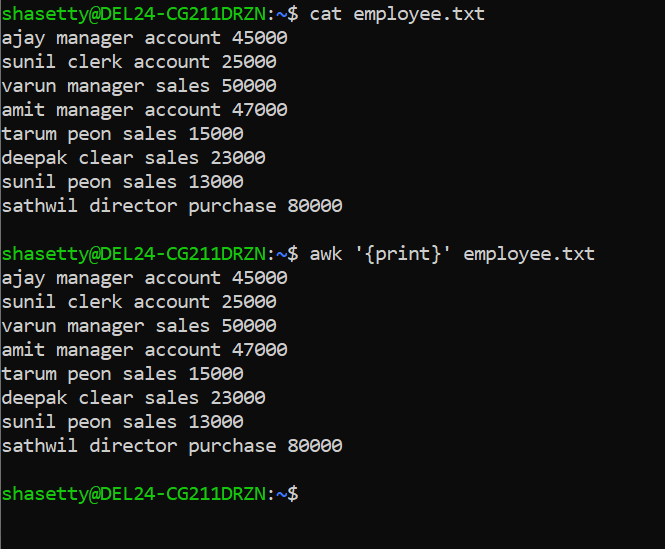


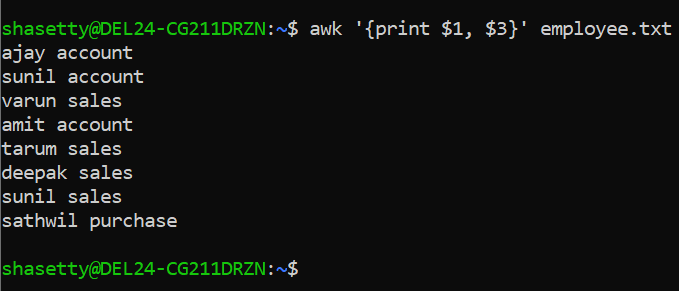


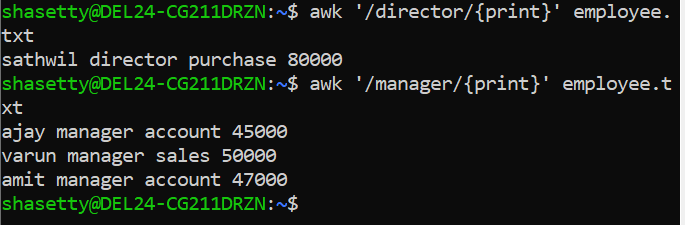


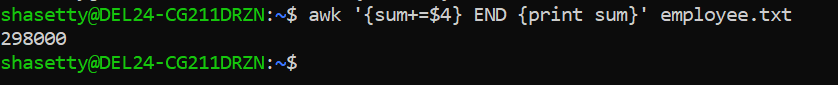
**TASK – 18**

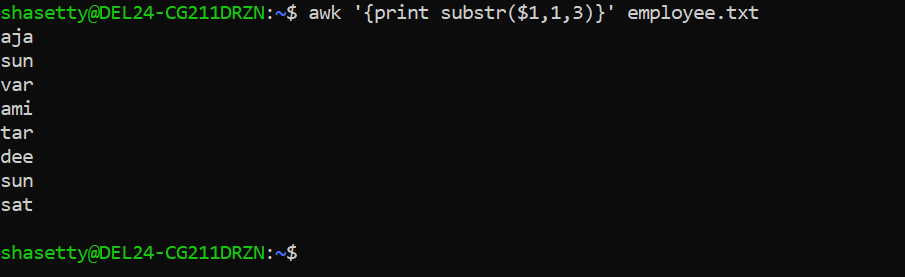
AWK commands in doc 15 Linux AWK commands.

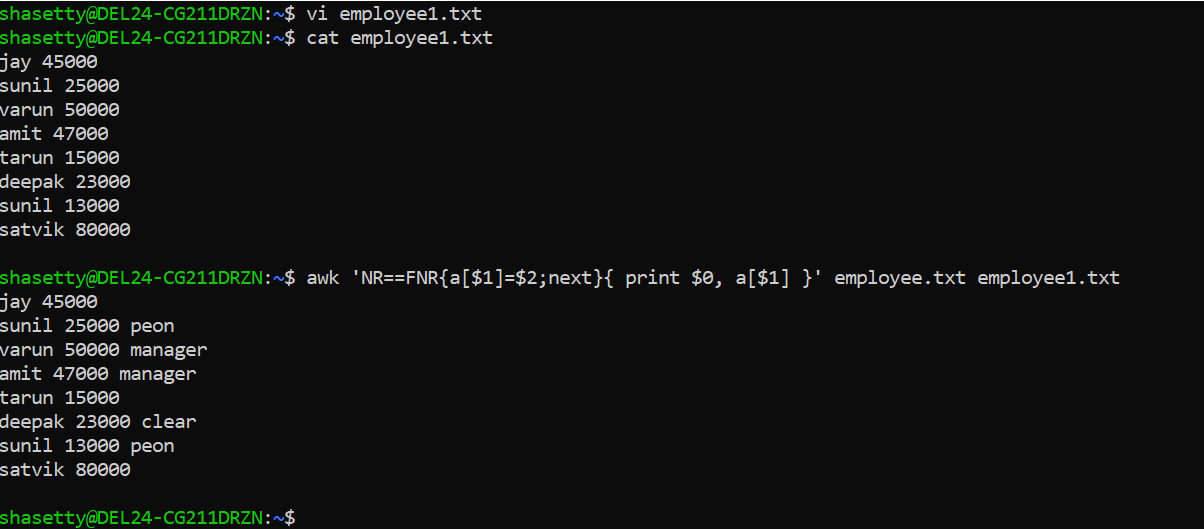


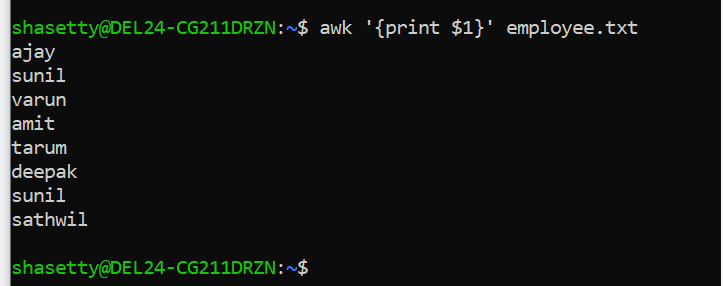


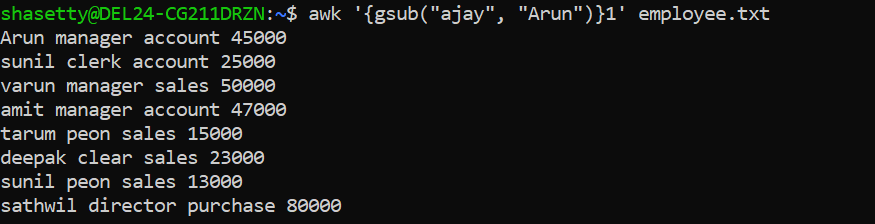


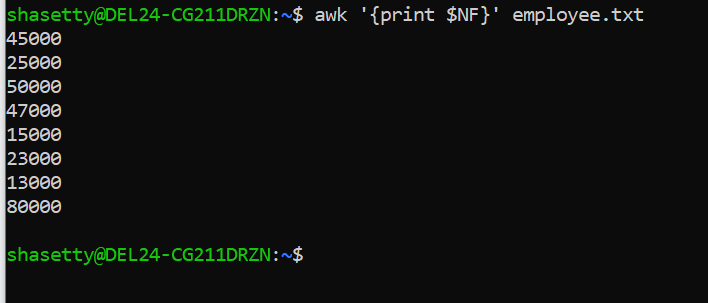


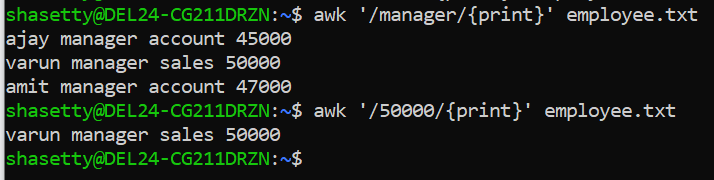


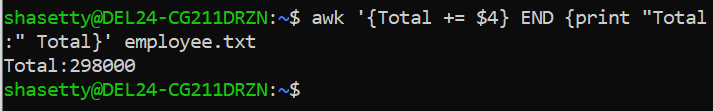


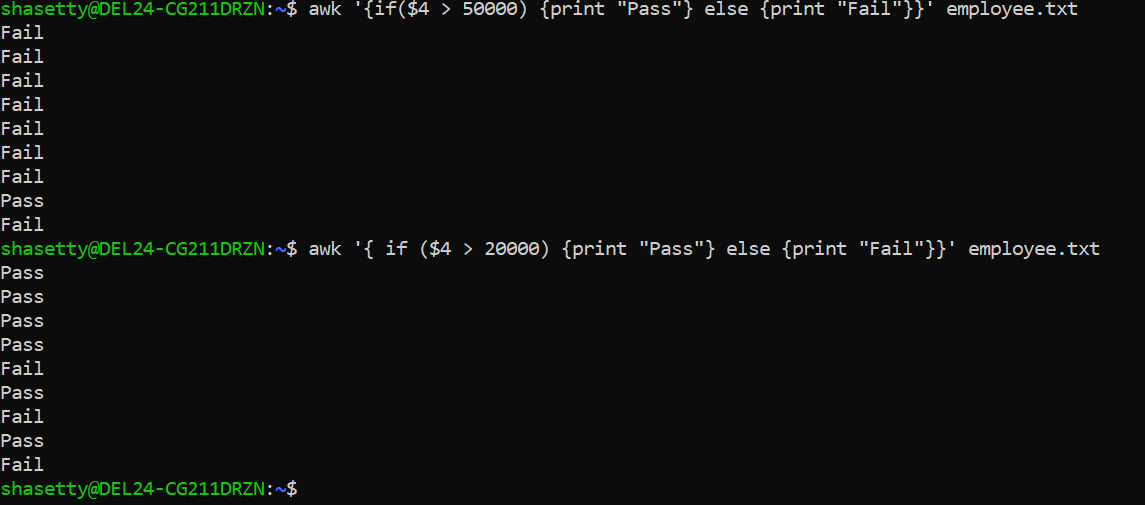


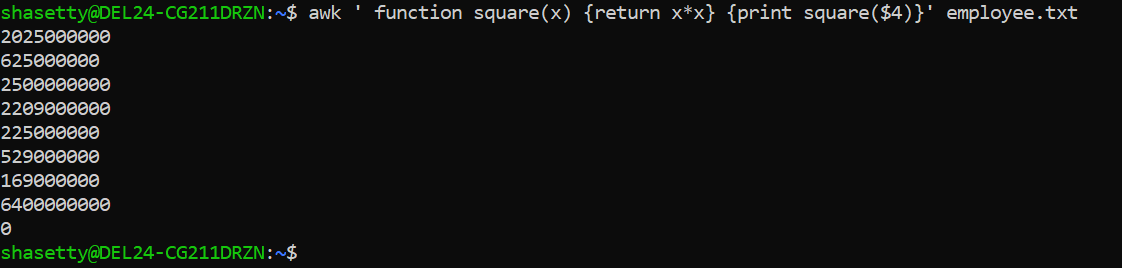






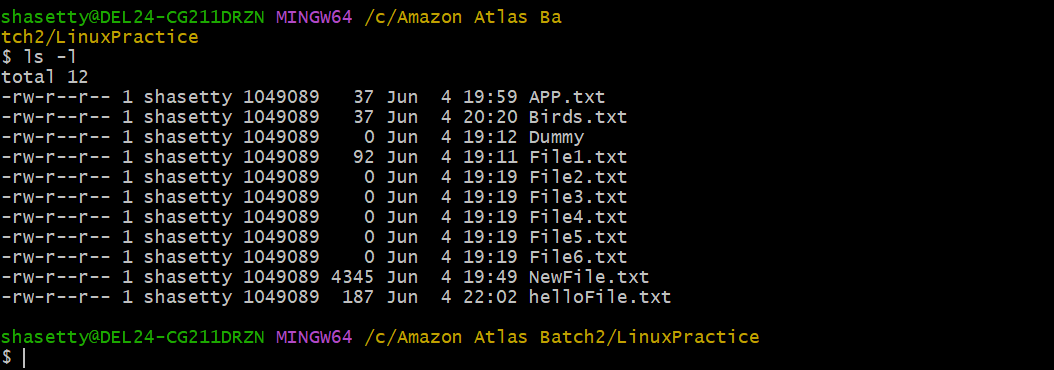






**TASK – 18**

How to check file access permission in Linux?



**TASK – 19**

What are the default permissions for a new file ?

Plz find out for

Owner   → ?

Group → ?

All and others → ?

When a **new file** is created, the default permissions are usually:

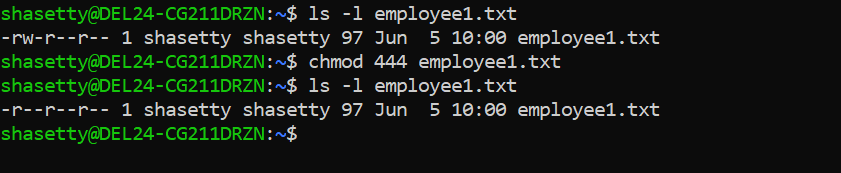
| **Category** | **Permission** |
| --- | --- |
| **Owner** | rw- |
| **Group** | r-- (can vary) |
| **Others** | r-- (can vary) |
| * **-rw-r--r--** | Basically it will be. |

This means:

* **Owner**: read & write (rw-)
* **Group**: read only (r--)
* **Others**: read only (r--)
* New files do not get execute (x) permission by default.
* We must manually add x if needed (e.g., using chmod +x for scripts).

**TASK – 20**

What is the command to change the permisssion to read only for the owner, group and all other users ?

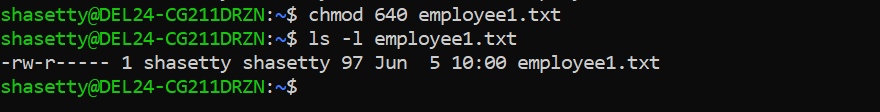


**TASK – 21**

Can you change the file permissions to match the following:

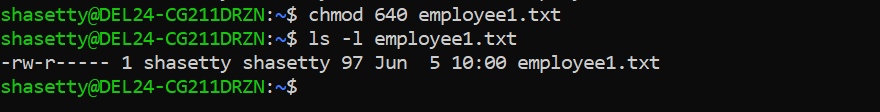
* owner: Read and Write
* group: Read

other: no permissions (None)



**TASK – 22**

What was the command for changing the file permissions to -rw-r-----?



**TASK – 23**

Change chmod.exercises permissions to -rwxr-x--x

Change the file permissions to match the following:

owner: Read, Write and Execute

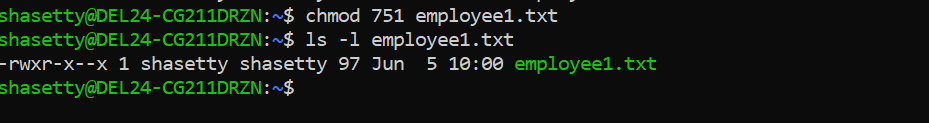
group: Read and Execute

other: Execute

 **Owner**: Read, Write, Execute → rwx = 7

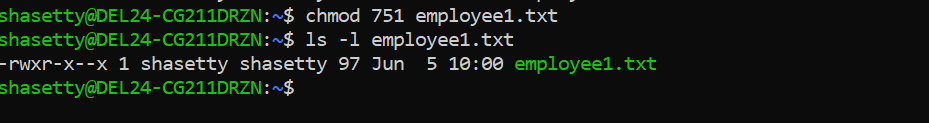
 **Group**: Read, Execute → r-x = 5

 **Others**: Execute only → --x = 1



**TASK – 24**

What was the command for changing the file permissions to -rwxr-x--x



**TASK – 25**

chown -c master file1.txt

**What does this command do ?**

 chown = change ownership of a file

 -c = report (print) changes made

 master = the new owner username

 file1.txt = the target file

this command **changes the ownership of** file1.txt **to the user** master, and **shows a message if it succeeded**.

**TASK – 26**

Can you define what is a process?

A **process** is an **instance of a running program**.

When you run any command, script, or application in a computer (like opening a browser or running ls in Linux), the operating system creates a **process** for that task.

**TASK – 27**

What is command to check foreground process and background process

A **foreground process** is the one that is currently running and interacting with your terminal (you can't type anything else until it's done or stopped).

**sleep 60**

This runs for 60 seconds **in the foreground** — you can’t use the terminal until it ends or is stopped.

**Background Process**

**sleep 60 &**

**jobs**

**ps -ef | grep sleep**

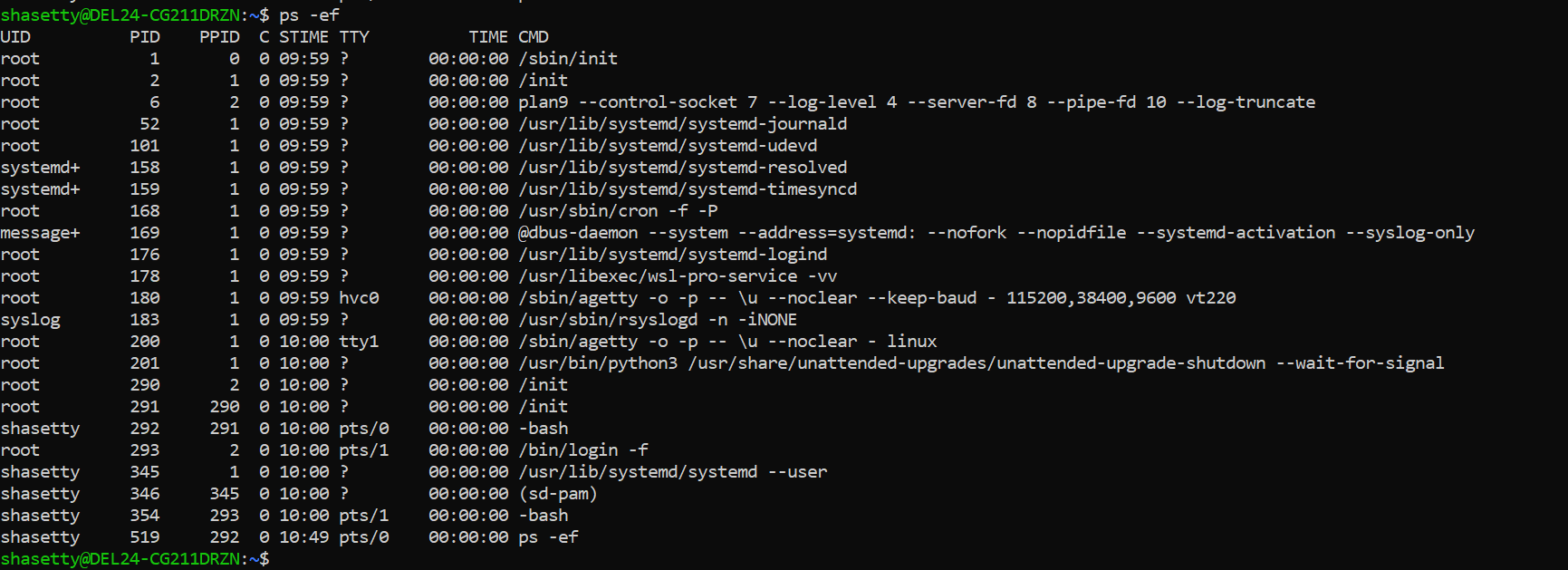
**fg %1**

**bg %1**

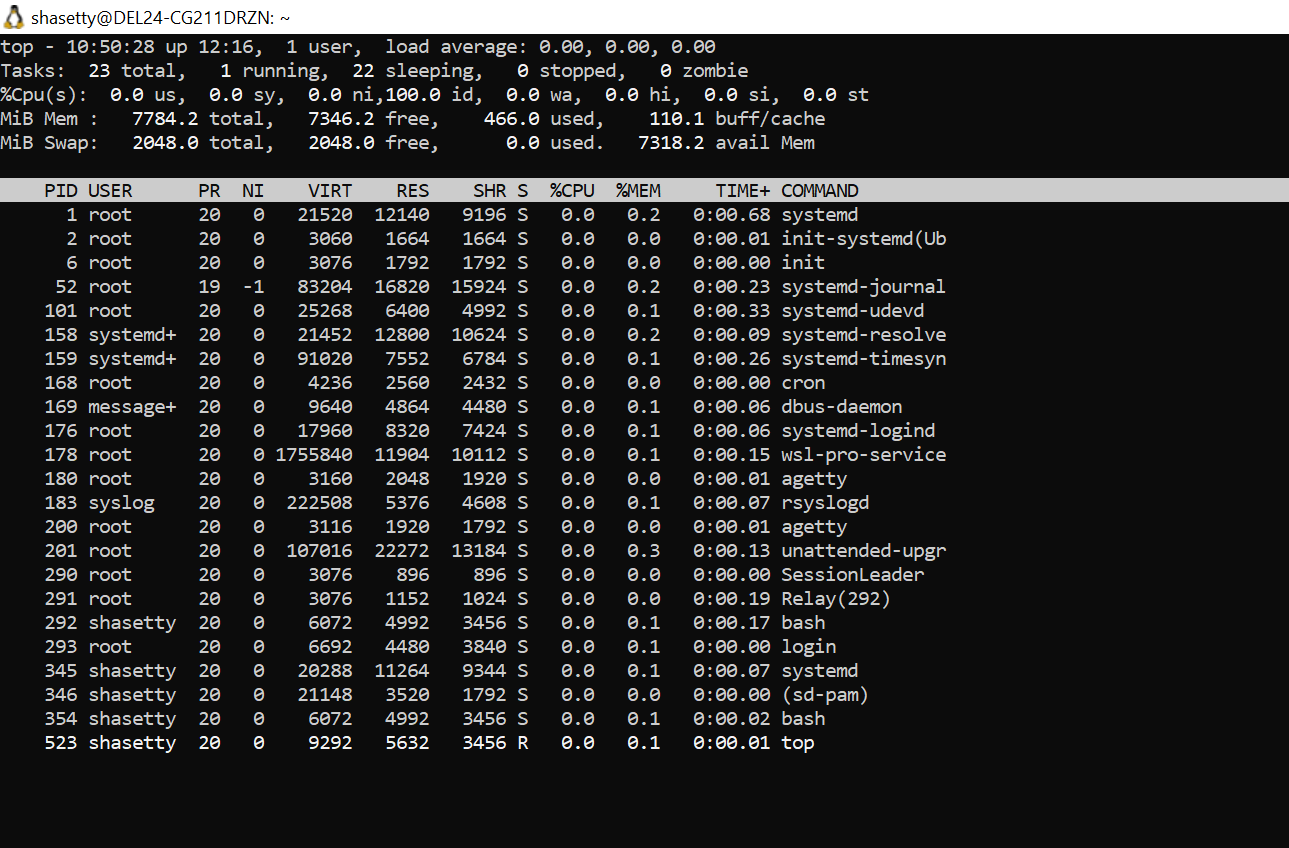
**TASK – 28**

Can you list all the running processes?

**ps -ef**

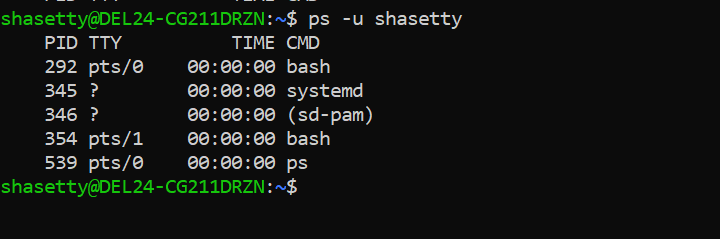


**Top -** Shows a **live, updating list** of all running processes

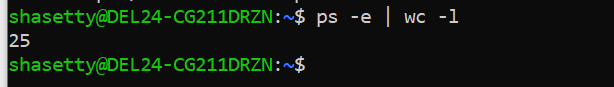


**Htop -**  Like top, but **easier to read**

To list only processes of a specific user:

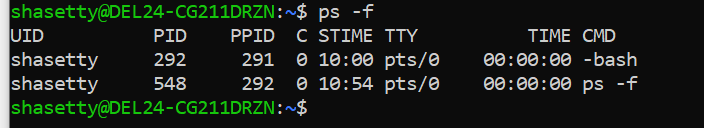


To count total running processes:



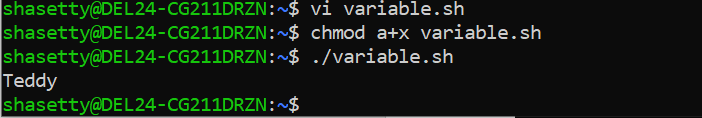
**TASK – 29**

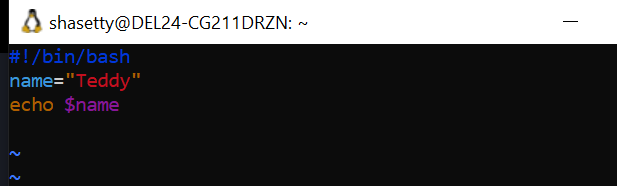
What will ps -f command do ? plz try n check .. ss required.



**TASK – 30**

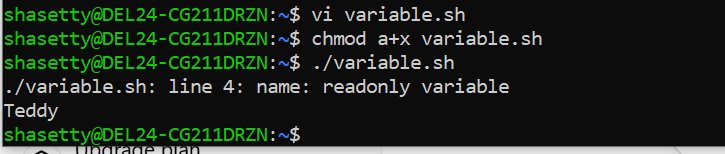
Can you create a variable name with your name in it ?





**TASK – 31**

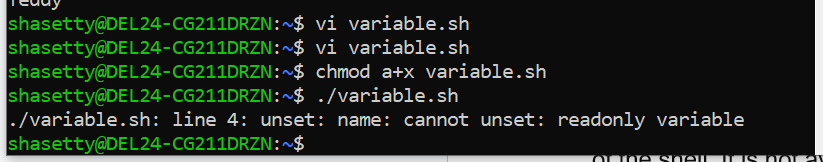
Can you make the above name variable read only..

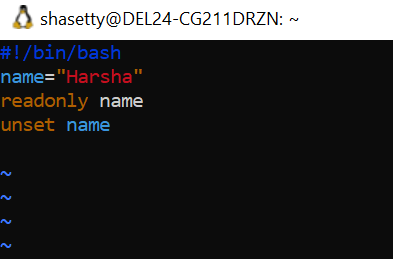




**TASK – 32**

Now will unset or delete the variables





**TASK – 33**

CAn u try to add a list of your friends names in an array and try to printout

Ex:

NAME[0]="Ram"

NAME[1]="Sita"

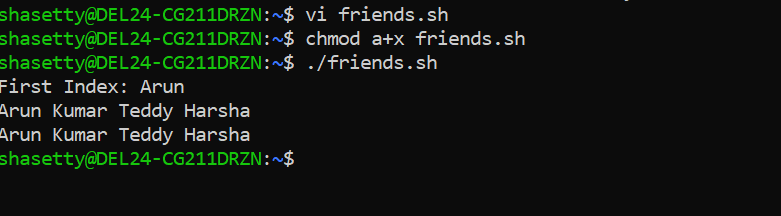
NAME[2]="Tina"

NAME[3]="Veena"

NAME[4]="Tim"

echo "First Index: ${NAME[0]}"

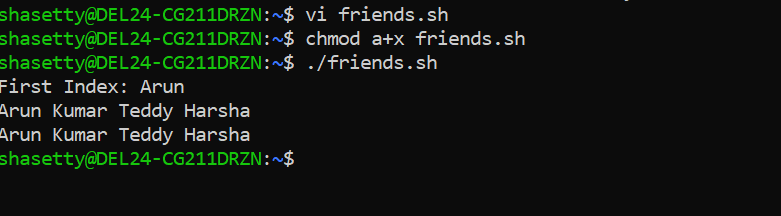
echo "Second Index: ${NAME[1]}"





**TASK – 34**

Can you print all the list at once in an array.. Try the below cmds and check





**TASK – 35**

a=0

while [ "$a" -lt 10 ]    # this is loop1

do

   b="$a"

   while [ "$b" -ge 0 ]  # this is loop2

   do

      echo -n "$b "

      b=`expr $b - 1`

   done

   echo

   a=`expr $a + 1`

Done

**OUTPUT**

**0**

**1 0**

**2 1 0**

**3 2 1 0**

**4 3 2 1 0**

**5 4 3 2 1 0**

**6 5 4 3 2 1 0**

**7 6 5 4 3 2 1 0**

**8 7 6 5 4 3 2 1 0**

**9 8 7 6 5 4 3 2 1 0**

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |